Stochastic Runtime Analysis of the Cross-Entropy Algorithm

Zijun Wu, Michael Kolonko, Rolf H. Möhring

Abstract—**This article analyzes the stochastic runtime of the cross-entropy algorithm for the well-studied standard problems OneMax and LeadingOnes.**

We prove that the total number of solutions the algorithm needs to evaluate before reaching the optimal solution (i.e. its runtime) is bounded by a polynomial $Q(n)$ in the problem size n with a probability growing exponentially to 1 with n if the parameters of the algorithm are adapted to n in a reasonable way. Our polynomial bound $Q(n)$ for OneMax outperforms the well-known runtime bound of the 1-ANT algorithm, a particular ant colony optimization algorithm. Our adaptation of the parameters of the cross-entropy algorithm balances the number of iterations needed and the size of the samples drawn in each iteration, resulting in an increased efficiency.

For the LeadingOnes problem, we improve the runtime of the algorithm by bounding the sampling probabilities away from 0 and 1. The resulting runtime outperforms the known stochastic runtime for a univariate marginal distribution algorithm, and is very close to the known *expected* runtime of variants of Min-Max Ant Systems. Bounding the sampling probabilities allows the cross-entropy algorithm to explore the search space even for test functions with a very rugged landscape as the LeadingOnes function.

Index Terms—**Cross-Entropy algorithm, 1-ANT algorithm, stochastic runtime analysis of evolutionary algorithms, stochastic complexity, analysis of randomized algorithms.**

I. Introduction

THE cross-entropy (CE) algorithm is a general purpose optimization tool that has been applied successfully to many $\mathcal{NP}$-hard combinatorial optimization problems, see e.g. the book [1] for an overview. It was initially designed for rare event simulation by Rubinstein [2] in 1997, and thereafter formulated as an optimization tool for both continuous and discrete optimization (see [3]).

CE has much in common with the famous ant colony optimization (ACO, see [4]) and the estimation of distribution algorithms (EDAs, see [5]). They all belong to the so-called *model-based search* paradigm (MBS), see [6]. Instead of only manipulating solutions, which is very typical in traditional heuristics such as genetic algorithms [7], local search [8] and others, MBS algorithms attempt to optimize the solution reproducing mechanism. In each iteration, they produce new solutions by sampling from a probability distribution on the search space. The distribution is often called a *model* in the literature (see e.g. [6] and [9]), which corresponds to the pheromone matrices in the literature of ACO algorithms. This model evolves iteratively by incorporating information from some elite solutions occurring in the search history, so as to asymptotically model the spread of optimal solutions in the search space. See the recent Thesis [9] for more details on MBS algorithms and their mathematical properties.

The present article concentrates on the theoretical analysis of CE for discrete optimization. We assume, without loss of generality, a problem (S, f) where S is a *finite* set of feasible solutions and $f : S \to \mathbb{R}$ is a *fitness function*. The objective is to find a feasible solution $s \in S$ maximizing the fitness function f. Specifying an initial model (distribution) $\mathbf{\Pi}_0$ on S, CE then repeatedly executes the following four steps in each iteration $t = 0, 1, 2, \ldots$, until a predefined stopping criterion is met (see Section II for more details):

Sampling: generate a random sample $\mathbf{X}_t \in S^N$ of a fixed sample size $N \in \mathbb{N}$ from the current model $\mathbf{\Pi}_t$;

Selection: sort these N solutions in $\mathbf{X}_t$ according to the fitness function f and select the best $M \in \mathbb{N}$ solutions from $\mathbf{X}_t$;

Learning: learn a new model (distribution) $\mathbf{W}_t$ from the selected solutions;

Update: construct the next model $\mathbf{\Pi}_{t+1}$ by a convex combination of $\mathbf{\Pi}_t$ and $\mathbf{W}_t$, i.e.,

$$\mathbf{\Pi}_{t+1} = (1 - \rho)\mathbf{\Pi}_t + \rho \mathbf{W}_t. \qquad (1)$$

Here, the initial model $\mathbf{\Pi}_0$ is typically the uniform distribution on S. The *sample size* N, the *elite size* M and the *smoothing parameter* $\rho \in (0, 1]$ are fixed parameters that need to be fine-tuned for practical use. ρ corresponds to the *evaporation rate* in ACO and *learning rate* in EDAs. In our case, the new model $\mathbf{W}_t$ is just the *empirical distribution* of the selected elite solutions which reflects the spread of high-quality solutions on S. By exploiting information from $\mathbf{W}_t$ (see (1)), the model $\mathbf{\Pi}_{t+1}$ may increase the probabilities of generating high-quality solutions in the next round. Thus, CE has some similarities with reinforcement learning [10].

We shall analyze the *stochastic complexity* of CE in discrete optimization. We measure the complexity by the number of solutions an algorithm has to sample before finding an optimal solution. As this number will be a random variable, we study the event $\mathcal{E}$ that the algorithm needs to sample at most $Q(n)$ solutions, for a bounding function $Q(n)$ depending on the

Z. Wu is with Beijing Institute for Scientific and Engineering Computing (BISEC), and also the Department of Applied Mathematics and Physics, Beijing University of Technology (BJUT), Pingleyuan 100, 100124, Beijing, P. R. China, e-mail: zijunwu1984a@163.com

M. Kolonko is with Institute for Applied Stochastic and Operations Research (IASOR), Technical university of Clausthal, Clausthal-Zellerfeld, D-38678, Germany, email: kolonko@math.tu-clausthal.de

R. H. Möhring is with Beijing Institute for Scientific and Engineering Computing (BISEC), Pingleyuan 100, 100124, Beijing, P. R. China, e-mail: rolf.moehring@tu-berlin.de

Submitted on Oct. 03, 2016; Revised and Resubmitted on Dec. 04, 2016; Accepted on Feb. 05, 2017.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2667713, IEEE Transactions on Evolutionary Computation

problem size n. We carefully adapt the parameters of the CE algorithm to the probability $\mathbf{P}(\mathcal{E})$ and are able to derive conditions under which $\mathbf{P}(\mathcal{E})$ grows exponentially fast to 1 with $n \to \infty$ for a given polynomial $Q(n)$.

Due to the famous No Free Lunch Theorem [11], we cannot derive the stochastic complexity of an algorithm for general problems. We therefore choose the simple *pseudo-boolean* problems ONEMAX and LEADINGONES as our test cases. For these problems, we put $S = \{0, 1\}^n$, then the fitness function $f(x)$ in ONEMAX simply counts the '1' in $x \in S$. Similarly, in LEADINGONES, $f(x)$ counts the consecutive leading '1' in x. Although these are only toy problems, the results actually help in understanding the efficiency of CE and can inspire parameter tuning for the practical problems. Their use is well-established in the literature. Section II gives a formal definition of ONEMAX and LEADINGONES.

So far, results on the asymptotic behavior of CE are mainly about its convergence, i.e., whether CE can find an optimal solution in finite time. Convergence analysis of CE was initiated by Costa et al. [12]. Wu and Kolonko [13] successfully extended the results of [12] to a more general framework. In particular, they proved that the '**Sampling**' step of CE will freeze at a single solution after a finite number of iterations (*stagnation*), and that the stagnant solution need not be optimal. This means that the well-know *genetic drift* phenomenon [14] also happens in CE and some ACO algorithms. Wu and Kolonko [13] also obtained an initial runtime result of CE for the LEADINGONES problem. These findings have been extended in [15] and [9] to a more general MBS framework.

Runtime analysis of evolutionary algorithms (EA) has gained rapidly growing interest in recent years, see e.g. [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], and [28]. In these papers the *runtime* of an algorithm is conventionally expressed by a bound on the total number of solutions evaluated before reaching an optimal solution in expectation or with high probability. Particularly, the dependence of this bound on the problem size n is studied. The first steps towards this type of runtime analysis of an evolutionary algorithm usually employ some simple pseudo-boolean functions as test problems, see e.g., [16], [20], [29], [22], [30]. Exceptions are some typical combinatorial optimization problems that have been studied in recent years, see e.g., the minimum spanning tree problem [31], and the traveling salesman problem [32]. We follow this convention, and inspect how efficient the general purpose evolutionary strategy CE is on ONEMAX and LEADINGONES.

Theoretical runtime analysis in the literature considers two cases: *expected runtime analysis* and *stochastic runtime analysis*. Expected runtime analysis inspects the average runtime of an algorithm on a particular problem, and can exploit mature probabilistic techniques such as *drift analysis* [17] and others [22]. Stochastic runtime analysis adds an overwhelming probability guarantee to the runtime result (see e.g. the classic runtime result of 1-ANT in [20]), and is thus much more informative than expected runtime analysis. Here, 'overwhelming' means a probability growing exponentially fast to 1. The stochastic runtime analysis is closely related to the randomized complexity class RP [33], which is of great importance in theoretical computer science. Since CE can stagnate with a single non-optimal solution, an expected runtime analysis makes only sense in special cases. We will therefore mostly deal with the stochastic runtime analysis of CE and just call it runtime in the sequel.

We employ the standard asymptotic notations from complexity theory to describe the dependence of the bounding function $Q(n)$ on the problem size n in the sequel. For a function $h(n)$ we use $O(h)$ to denote the class of functions g which are *bounded from above* by h, i.e., $g(n) \leq c \cdot h(n)$, $\forall n \geq n_0$, for some constant $c > 0$ not depending on n, and some $n_0 \in \mathbb{N}$. Similarly, $\Omega(h)$ is the class of functions g that are *bounded from below* by h, i.e., there exists $n_0 \in \mathbb{N}$ and $c > 0$ not depending on n such that $g(n) \geq c \cdot h(n)$, $\forall n \geq n_0$. Class $\Theta(h)$ is the intersection of $\Omega(h)$ and $O(h)$. In addition, class $o(h)$ denotes the class of functions g with $g(n)/h(n) \to 0$ as $n \to \infty$, and class $\omega(h)$ denotes the class of functions g with $g(n)/h(n) \to \infty$ as $n \to \infty$.

EAs that have been extensively studied for their runtimes are theoretical abstractions of the Max-Min Ant System (MMAS, see [34]) like 1-ANT [18], MMAS$_{bs}$ [22], (1+1) MMAA [29], and those algorithms derived from the genetic algorithm [7] like (1+1) EA [16] and univariate marginal distribution algorithm (UDMA, see [30]). Some more details on these algorithms are given at the end of Section II. Runtime results of EAs for ONEMAX can be found in e.g. [16], [18], [20], [22], [29], [35] and [36]. Droste et al. [16] showed that (1+1) EA has an expected runtime of $\Theta(n \ln n)$ on ONEMAX. Neumann and Witt [18] and [20] thoroughly analyzed the stochastic runtime of the classic 1-ANT algorithm on ONEMAX. They showed that, if $\rho = \Omega(n^{-1+\epsilon})$ for some constant $\epsilon \in (0, 1)$, then the runtime of 1-ANT is $O(n^2)$ with probability $1 - 2^{-\Omega(n^{\epsilon/2})}$. But if $\rho = O(n^{-1-\epsilon})$ for a constant $\epsilon > 0$, then the runtime of 1-ANT is $2^{\Omega(n^{\epsilon/3})}$ with probability $1 - 2^{-\Omega(n^{\epsilon/3})}$. Doerr and Johannsen [35] continued the study of [18] and [20]. They investigated the runtime of 1-ANT on ONEMAX with ρ in the critical window $(n^{-1-\epsilon}, n^{-1+\epsilon})$, and proved that the stochastic runtime of 1-ANT is super-polynomial when $\rho = o(\frac{1}{n \ln n})$. Gutjahr and Sebastiani [22] showed that in the case of a constant *reward* (i.e. the amount of pheromones an ant deposits on an arc), the expected runtime of MMAS$_{bs}$ on ONEMAX is $O(n \ln n)$ when $\rho = \Omega(1)$, and $O(n^2 \ln n)$ when $\rho = \frac{1}{n}$. In the case of a fitness-proportional reward, Gutjahr and Sebastiani further showed in [22] that the expected runtime of MMAA$_{bs}$ on ONEMAX is $O(n(\ln n)^2)$, when $\rho = \frac{1}{n}$. Neumann et al. [29] compared different abstractions of the MMAS algorithm [34] that are frequently employed in runtime analysis, and analyzed the expected runtime of MMAS$_{bs}$ and (1+1) MMAA on unimodal problems with the technique proposed in [22]. In particular, they proved that the expected runtime of MMAS$_{bs}$ on ONEMAX is $O(\frac{n \ln n}{\rho})$ which extends the result in [22]. More recent runtime results on ONEMAX were obtained for the so-called $(1 + (\lambda, \lambda))$ EA in [36] which states that if the offspring size is $\lambda \in [1, n]$ and the mutation probability is λ/n, then the expected runtime of $(1 + (\lambda, \lambda))$ EA on ONEMAX is $O(\max\{n \log n/\lambda, n\lambda \log \log n/ \log(n)\})$.

Runtime results of EAs for LEADINGONES can be found in e.g. [22], [29], [21], [30] and [13]. Gutjahr and Sebastiani [22] proved that the expected runtime of MMAS$_{bs}$ on LeadingOnes is bounded from above by $n[\frac{\log\delta_{\min}+\log(1-\delta_{\min})}{\log(1-\rho)}]+e\cdot n^2$, where $e = 2.718\dots$ is the Euler's number, $\delta_{\min}$ is a fixed probability lower bound, which is often set to be $\frac{1}{n}$ in the literature. Neumann et al. [29] further proved that the expected runtime of MMAS$_{bs}$ is $O(n^2 + (n\ln n)/\rho)$. Doerr et al. [21] showed that when $\rho \in o(1/(n\ln n))$, then the runtime of 1-ANT on LeadingOnes is super-polynomial with overwhelming probability. Moreover, Doerr et al. [21] also showed that the expected runtime of 1-ANT on LeadingOnes is $O(n^2(6e)^{1/(n\rho)})$. Chen et al. [30] proved that by taking $N = \omega(n^{2+\epsilon}\ln n)$ and $M = \Theta(N)$, then UMDA with truncated selection finds the optimal solution of LeadingOnes with overwhelming probability in $O(n)$ steps. Wu and Kolonko [13] showed that CE with parameter ρ bounded away from 0 needs an *infinite expected runtime* for any non-trivial combinatorial optimization problem due to absorption in possibly non-optimal solutions, see also [15]. They also proved that by taking $N = \Omega(n^{2+\epsilon}), \rho = \Omega(1)$ and $M = \beta N$ for a specified $\beta \in (0,1)$ *depending* only on ρ, CE finds the optimal solution of LeadingOnes with overwhelming probability in n iterations.

Up to now, the results from [13] seem to be the only general runtime results for CE in the literature. The present article continues and considerably extends this study. Our new results are given in Section III in Theorem 1 for the OneMax problem and in Theorem 2 for the LeadingOnes problem. More precisely, in Theorem 1 on OneMax we assume that the parameters of CE depend on the problem size n in such a way that $\rho = \rho(n)$ is bounded away from 0, $M = M(n) = \Omega(n^{1+\epsilon})$ for some $\epsilon \in (0,1)$, and $N = N(n) = \omega(M)$. Then, with an overwhelming probability of at least $1 - e^{-\Omega(n^{\epsilon/3})}$, CE finds the optimal solution in $O(n^{0.5+\frac{1}{3}\epsilon}N/\rho)$ steps (number of iterations). If $N = n^{1+\epsilon}\ln n$ and $\rho \in \Omega(1)$, the bounding polynomial $Q(n) := n^{0.5+\frac{1}{3}\epsilon}N/\rho = n^{1.5+\frac{4}{3}\epsilon}(\ln n)$ shows a much better asymptotic behavior than n^2. Hence our result improves the best known stochastic runtime $O(n^2)$ of 1-ANT on OneMax reported in [18] and [20].

In other words, if the sample size N and the elite size M are moderately adapted to the problem size n, and if the smoothing parameter ρ is not exponentially small, then CE reaches the optimal solution of OneMax in a polynomially bounded runtime with high probability. Theorem 1 also shows how the magnitude of the smoothing parameter ρ affects the runtime: the larger the smoothing parameter ρ, the smaller the total number of *iterations* CE needs to find the optimal solution. This underlines a finding of [3] and [37] that a moderately large sample size per iteration and a relatively large ρ allow fewer iterations and a better overall runtime. Although the expected runtime of CE is infinite, the distribution of the runtime is actually biased heavily towards small values for this particular problem.

Theorem 2 gives our results on the LeadingOnes problem, extending results from [13]. We modified the basic CE algorithm by introducing upper and lower bounds on the sampling probabilities as known from MMAS. We are then able to show, that with ρ bounded away from 0, $N = \Omega(n^{1+\epsilon})$ and $M = O(n^{\epsilon/2})$ $(M \geq 1)$ for an arbitrary $\epsilon > 0$, the runtime is $O(n^{2+\epsilon})$, with probability at least $1 - e^{-\Omega(n^{\epsilon})}$. This result is very close to the best known *expected* runtime $O(n^2)$ that was shown for different variants of MMAS, see e.g. [21], [29] and [22].

Different from OneMax, the LeadingOnes problem has dependencies between bits and is often used as a standard test problem with a rugged search space. The improvement of the new bound $O(n^{2+\epsilon})$ over the bound $O(n^{3+\epsilon})$ from [13], shows that bounding the sampling distribution away from its extreme values 0 and 1 allows CE to explore the search space more globally and more efficiently.

The remainder of this article is arranged as follows. Section II formally defines the CE algorithm, Section III analyses the stochastic runtime of our two test problems OneMax and LeadingOnes. A short conclusion to the whole article and a suggestion for possible future work is given in Section IV. Interested readers are referred to the Appendix for detailed proofs of our results.

II. The Algorithm

CE has slightly different definitions for different problems, see e.g. [1], [12] and [13]. We employ here the basic definition of CE for discrete optimization from [12]. As we consider OneMax and LeadingOnes, we assume a *maximization* problem (S, f) with search space $S = \{0,1\}^n$ (i.e. a product space over $\{0,1\}$) for some problem size $n \in \mathbb{N}$ and a *fitness function* $f : S \mapsto \mathbb{R}$. We consider only product distributions as the models for sampling. In the literature of MBS, product distributions are also called univariate marginal models (UMM, see, e.g., [9] and [5]). Algorithm 1 formally defines the CE algorithm, its details are explained in the following.

In the original version of CE (see [3]), the learning step chooses as $\mathbf{W}_t$ that distribution from a given set of feasible models that is closest to the uniform distribution on the selected elite part of the sample. Here, the distance between distributions is measured by the cross entropy, a concept often used in information theory. In our case with $S = \{0,1\}^n$, the empirical distribution from the selected solutions minimizes the cross entropy among all product distributions on S.

In Algorithm 1, a product distribution is written as a vector $\boldsymbol{\Pi} = (p_1, \dots, p_n) \in [0,1]^n$, where p_i is the probability of producing bit '1' at position i for each $i = 1, \dots, n$. A solution $s = (s_1, \dots, s_n) \in S$ is sampled from $\boldsymbol{\Pi}$ with probability

$$\mathbf{p}(s \mid \boldsymbol{\Pi}) = \prod_{i=1}^{n} p_i^{s_i}(1-p_i)^{1-s_i}. \tag{3}$$

In each iteration, CE biases the model towards the best solutions of the present sample in the hope to approximate the maximizing model $\boldsymbol{\Pi}^*$ of the optimization problem, i.e.,

$$\max_{\boldsymbol{\Pi}\in[0,1]^n} \mathbf{p}(S^* \mid \boldsymbol{\Pi}), \tag{4}$$

where $S^* \subset S$ denotes the set of optimal solutions.

Initially, we have to select a model $\boldsymbol{\Pi}_0 = (p_{0,1}, \dots, p_{0,n})$ which we require to be *strictly positive*, i.e., $p_{0,i} \in \Omega(1)$ for all $i = 1, 2, \dots, n$. A typical example for $\boldsymbol{\Pi}_0$ is the *uniform*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2667713, IEEE Transactions on Evolutionary Computation

```
Algorithm 1 The Cross-Entropy algorithm
Require:
    an initial model $\mathbf{\Pi}_0 = (p_{0,1}, \ldots, p_{0,n})$ with each entry $p_{0,i} \in (0,1)$, a fixed smoothing parameter $\rho \in (0,1]$ and a fixed sample size $N \in \mathbb{N}$, a fixed elite size $0 < M \le N$.
t = 0;
loop
    draw a random sample $\mathbf{X}_t = (X_t^{(1)}, \ldots, X_t^{(N)})$ of size $N$ from the current model $\mathbf{\Pi}_t$;
    sort these $N$ solutions according to the fitness function $f$ as $X_t^{[1]}, \ldots, X_t^{[N]}$ such that
        $f(X_t^{[1]}) > f(X_t^{[2]}) > \cdots > f(X_t^{[N]}),$
    where we use the convention that if $f(X_t^{(j)}) = f(X_t^{(l)})$ we write $f(X_t^{(j)}) > f(X_t^{(l)})$ provided that $j > l$;
    calculate an empirical model $\mathbf{W}_t = (w_{t,1}, \ldots, w_{t,n})$ from the selected elite sample
        $(X_t^{[1]}, \ldots, X_t^{[M]})$
    with each
        $w_{t,i} := \frac{\sum_{m=1}^{M} X_t^{[m]}(i)}{M}$        (2)
    where we write $X_t^{[m]} = (X_t^{[m]}(1), \ldots, X_t^{[m]}(n))$ for all $m = 1, \ldots, N$;
    $\mathbf{\Pi}_{t+1} = (1-\rho)\mathbf{\Pi}_t + \rho \cdot \mathbf{W}_t$;
    $t = t + 1$;
end loop
```

distribution on S, in which $p_{0,i} = 1/2$ for each $i = 1, 2, \ldots, n$. Of course, if prior information is available, one may employ a biased distribution as the initial model, or incorporate the prior information into the sampling in each iteration, as has been frequently done in ACOs [13].

In each iteration $t \in \mathbb{N}$, a random sample $\mathbf{X}_t$ of a fixed size N is independently drawn from $\mathbf{\Pi}_t$ where each random solution $X_t^{(j)} := (X_t^{(j)}(1), \ldots, X_t^{(j)}(n)) \in S$ in $\mathbf{X}_t$ is sampled with a probability $\mathbf{P}(X_t^{(j)} = s) = \mathbf{p}(s|\mathbf{\Pi}_t)$ as defined in (3). By a *truncated* selection, M 'elite' solutions $X_t^{[1]}, \ldots, X_t^{[M]}$ are selected to build the empirical model $\mathbf{W}_t$ (see (2)), where $M < N$ is a fixed elite size. $\mathbf{W}_t$ is an approximation to the uniform distribution on the truncated search space $\{s \in S : \ f(s) \ge f(X_t^{[M]})\}$, and thus models the region of high-quality solutions.

The model $\mathbf{\Pi}_{t+1} = (p_{t+1,1}, \ldots, p_{t+1,n})$ for the next sampling is set to be a convex combination of the old model $\mathbf{\Pi}_t$ and the empirical model $\mathbf{W}_t$. In more detail, each component $\mathbf{\Pi}_{t+1,i}$ is obtained as

$$p_{t+1,i} = (1-\rho)p_{t,i} + \rho w_{t,i} \tag{5}$$

for $i = 1, 2, \ldots, n$, where $w_{t,i}$ is calculated by (2), and $\rho \in (0,1]$ is a fixed smoothing parameter. The parameter ρ plays a crucial role in the convergence analysis of CE algorithms, see [12] and [13]. Intuitively, a large ρ makes the next sampling concentrate more on the space spanned by present elite solutions and therefore will put more emphasis on *exploiting* this particular local region in the next sampling. On the other hand, a small ρ gives more chances to solutions that are outside that particular region and hence saves the potential for *global exploration*.

The stagnation behavior mentioned above means that the models $(\mathbf{\Pi}_t)_{t=0}^{\infty}$ converge to a one-point model concentrating on a particular solution with probability 1 as an approximation solution to (4). Note that a one-point model in this case coincides with the solution on which it concentrates.

At the end of this Section, we give a brief description of those algorithms from the literature that seem to be most similar to CE. These are 1-ANT, MMAS$_{bs}$, (1+1) MMAA and UDMA. 1-ANT, MMAS$_{bs}$ and (1+1) MMAA are rather similar to each other. They produce *exactly one* solution per iteration and update the model (with the best solution found so far) only when a better solution has been found. The time until an update is thus random, and so they resemble Algorithm 1 with a *dynamic sample size*. See [29] for a more detailed comparison between these three algorithms. Another significant difference to Algorithm 1 is that they favor global exploration by preserving a large diversity of solutions. In more detail, after applying the update (5), they will further adjust the model $\mathbf{\Pi}_{t+1} = (p_{t+1,1}, \ldots, p_{t+1,n})$ for the next iteration by

$$p_{t+1,i} = \begin{cases} p_{t+1,i} & \text{if } p_{t+1,i} \in [\delta_{\min}, \delta_{\max}], \\ \delta_{\min} & \text{if } p_{t+1,i} < \delta_{\min}, \\ \delta_{\max} & \text{if } p_{t+1,i} > \delta_{\max}, \end{cases} \tag{6}$$

$i = 1, 2, \ldots, n$, where $\delta_{\min}, \delta_{\max} \in (0,1)$ are constant lower and upper bounds that are often set to be $\frac{1}{n}, 1 - \frac{1}{n}$, respectively. As was summarized in the introduction, these algorithms are rather efficient for the standard test problems ONEMAX and LEADINGONES, and also some simple TSP instances in [24]. We will show in the next Section that Algorithm 1 outperforms these algorithms with respect to the total number of solutions sampled, even if CE is allowed to take larger samples per iteration.

Finally, UMDA is a particular EDA algorithm [5]. It can be seen as a radical version of CE in which $\rho = 1$, i.e., updates in the next iteration completely stay in the space spanned by the elite solutions of the previous iteration. Moreover, UMDA may employ more versatile selection mechanisms to choose solutions for estimating $\mathbf{W}_t$, e.g., truncated selection, fitness based random selection, purely random selection, and others. The runtime of UDMA on LEADINGONES was studied in [30] for truncated selection (a special case of Algorithm 1). Section III will show furthermore that in this case the runtime is significantly reduced when (6) is applied.

III. MAIN RESULTS

As mentioned above, the runtime of an evolutionary algorithm is conventionally expressed as the total number of evaluated solutions before an optimal solution is reached. Hence, the runtime of Algorithm 1 with sample size N equals $N \cdot \tau$, where $\tau := \min\{t \in \mathbb{N} : \ \mathbf{X}_t \cap S^* \ne \emptyset\}$ with $\min \emptyset = +\infty$ denotes the necessary number of iterations. *Stochastic runtime analysis* studies the probability $\mathbf{P}(N \cdot \tau < Q(n))$ for some bounding function $Q(n)$ depending on the problem size n. In particular, under a polynomial sample size N, we are interested in polynomial bounds $Q(n)$ for which $\mathbf{P}(\tau < Q(n))$ is growing exponentially fast to 1 as n increases.

A. A universal result on the expected runtime of CE

[13] showed that for any combinatorial optimization instance (S, f) with $S^* \neq S$, if $\rho > 0$, there is a positive probability that Algorithm 1 never sees an optimal solution. This yields the following Lemma.

Lemma 1. *The expected runtime of Algorithm 1 on any non-trivial combinatorial optimization problem ($S \neq S^*$) is infinite.*

Proof: See [13] for a proof. ■

Lemma 1 points out a possible weakness of Algorithm 1, namely the lack of global exploration power. Therefore, in the runtime analysis of CE, one needs to efficiently compensate for global exploration. The stochastic runtime results in Theorem 1 and Theorem 2 will pave a way for this.

B. Stochastic runtime results for OneMax

OneMax has the fitness function

$$f(s) = \sum_{i=1}^{n} s_i, \quad s = (s_1, \ldots, s_n) \in S = \{0,1\}^n. \tag{7}$$

It has an unique optimal solution $(1, 1, \ldots, 1)$. OneMax was introduced in [38] where Mühlenbein showed that the expected runtime of the classic (1+1) EA for this problem is $O(n \ln n)$. Since then, OneMax is used as a simple benchmark in runtime analysis of general-purposed heuristics that use an intelligent and self-adaptive search. Note that it is always worthwhile to check whether a general-purpose heuristic does perform well for such a simple problem as OneMax.

Theorem 1 lists our stochastic runtime results of Algorithm 1 on OneMax. The results show (see Theorem 1 b1)) that Algorithm 1 may *beat* the famous 1-ANT, for which the best known stochastic runtime is $O(n^2)$. Moreover, Theorem 1 b) shows that whenever ρ is also adapted to the problem size but not decreasing exponentially fast, then the stochastic runtime of Algorithm 1 is still polynomial. This is rather different from the case of 1-ANT, for which [35] showed that the stochastic runtime of 1-ANT is already super-polynomial when $\rho \in o(\frac{1}{n \ln n})$. A possible reason may be that 1-ANT needs a large ρ to record enough good information from elite solutions found, so as to efficiently exploit the neighborhood of these solutions in future iterations. By employing a large sample size, Algorithm 1 can weaken the role of ρ in generating better solutions (in future iterations) to a certain degree, and is therefore not so sensitive to the size of ρ compared to 1-ANT.

Theorem 1 (Runtime results for OneMax)**.** *Let $\epsilon \in (0, 1)$ be any small constant that is independent of n and all other parameters of the algorithm, and assume $p_{0,i} \in \Omega(1)$ for all $i = 1, \ldots, n$.*

a) *Assume that the elite size $M \in \Omega(n^{1+\epsilon})$, the sample size $N \in \omega(M)$. Then, with probability at least $1 - 2^{-\Omega(n^{\epsilon/3})}$, we have for τ, the* number of iterations *Algorithm 1 needs to find the optimal solution for* OneMax*, that*

$$\tau \leq (\lceil \tfrac{1}{4\epsilon/3} \rceil + 1) \frac{n^{1/2+\epsilon/3}}{\rho}.$$

b) *Assume that $M = n^{1+\epsilon}$, $N = \frac{1}{\lceil \frac{1}{4\epsilon/3} \rceil + 1} n^{1+\epsilon} \ln n$. Then a) holds and the stochastic runtime of Algorithm 1 is $O(Q(n))$ with probability at least $1 - 2^{-\Omega(n^{\epsilon/3})}$, where the bound $Q(n)$ is given by*

$b1$) $Q(n) = n^{1.5 + \frac{4}{3}\epsilon} \ln n$ *if $\rho \in \Omega(1)$,*
$b2$) $Q(n) = n^2$
if $0 < \epsilon < \frac{3}{8}$, and if $\rho \in \Omega(n^{-0.5 + \frac{4}{3}\epsilon} \ln n)$,
$b3$) $Q(n) = n^{1.5 + d + \frac{4}{3}\epsilon} \ln n$
if for some constant $d > 0$, $\rho \in \Omega(n^{-d})$.

For a proof of Theorem 1 a), see the Appendix. Theorem 1 b) is a direct consequence of a). In the proof, we actually show that the *elite solutions* converge to the optimal solution within polynomially many iterations with an overwhelming probability. Therefore, Theorem 1 $b1$) shows further that for $\rho = 1$ the sampling will freeze with an overwhelming probability at the optimal solution within $O(n^{0.5 + \frac{1}{3}\epsilon})$ iterations, and the models will converge to the optimal solution. This theoretically supports the motivation of CE as expressed by (4).

C. Runtime results for LeadingOnes

The LeadingOnes problem was proposed in [39] where Rudolph proved that the (1+1) EA has an expected runtime of $O(n^2)$. LeadingOnes is a more difficult problem compared to OneMax. Formally, its fitness function can be defined as

$$f(x) = \sum_{i=1}^{n} \prod_{j=1}^{i} x_i \quad x = (x_1, \ldots, x_n) \in S = \{0,1\}^n, \tag{8}$$

where n is again the problem size. Obviously, it also has the same unique optimal solution $(1, 1, \ldots, 1)$ as OneMax. However, it has dependencies between different positions, the fitness value does not only depend on the total number of ones but also on the position of the ones and therefore possesses a more difficult landscape than OneMax.

Theorem 2 lists our stochastic runtime results of Algorithm 1 on LeadingOnes. Theorem 2 a) generalizes the result of [13] by removing the dependency of the elite size M on the smoothing parameter ρ. Theorem 2 b) shows a much shorter stochastic runtime of Algorithm 1 by employing probability bounds as were used for MMAS$_{bs}$ and 1-ANT.

Theorem 2 (Runtime results for LeadingOnes)**.** *Let $\epsilon \in (0, 1)$ be a small constant that is independent of n and all other parameters of the algorithm and let $\mathbf{\Pi}_0$ be a strictly positive starting distribution.*

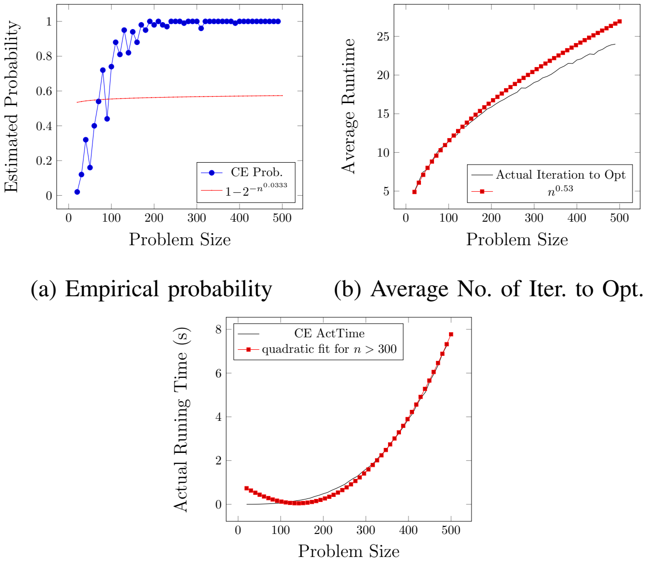
a) *If $M = \Omega(n^{2+\epsilon})$ and $N \in \omega(M)$, then for any $\rho \in \Omega(1)$ we have for τ, the total number of iterations needed by Algorithm 1 to reach an optimal solution, that $\tau \leq n$ with an overwhelming probability $1 - e^{-\Omega(n^{\epsilon})}$, i.e., the stochastic runtime in this case is $O(N \cdot n)$.*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2667713, IEEE Transactions on Evolutionary Computation

b) *If Algorithm 1 uses the update step* (6) *with lower bound* $\delta_{\min} = \frac{1}{n}$ *and upper bound* $\delta_{\max} = 1 - \frac{1}{n}$ *on the probabilities, then for* $N = \Omega(n^{1+\epsilon})$, $M \in O(n^{\epsilon/2})$ *and any* $\rho \in \Omega(1)$, *we have for* τ, *the total number of iterations needed by Algorithm 1 to reach an optimal solution, that* $\tau \le n$ *with overwhelming probability* $1 - e^{-\Omega(n^{\epsilon})}$. *Therefore, if we take* $N = n^{1+\epsilon}, M = O(n^{\epsilon/2})$, *the stochastic runtime of Algorithm 1 is* $O(n^{2+\epsilon})$.

For a proof of Theorem 2, see the Appendix. From Theorem 2, one can see that Algorithm 1 may take more samples to find the optimal solution of LEADINGONES than for the simpler ONEMAX. However, it still finds the optimal solution in polynomial time with an overwhelming probability. The results are attained under a relatively large smoothing parameter $\rho \in \Omega(1)$. This may indicate that a moderately large sample size N and a large ρ enable Algorithm 1 to find optimal solutions efficiently, even in the case of a difficult search space. Moreover, Theorem 2 b) may indicate that in this case employing probability bounds as in (6) may be very helpful.

D. Empirical study



(a) Empirical probability (b) Average No. of Iter. to Opt.

(c) Actual time

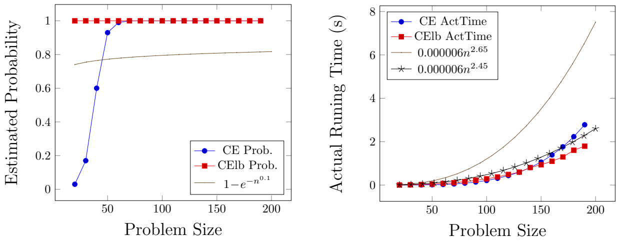
Fig. 1: Empirical study of CE for ONEMAX

1) Empirical study of ONEMAX*:* For the empirical study of CE with ONEMAX, we used the following parameter settings: $p_{0,i} \equiv 0.5$ (i.e. uniform distribution as Π_0), $\rho = 0.9$. We put $\epsilon = 0.1$, $M = \lceil \frac{n^{1.1}}{15} \rceil, N = M \cdot \lceil n^{0.6} \rceil$. Hence the assumptions of Theorem 1 a) are fulfilled, the upper bound for τ becomes $(\lceil \frac{1}{0.4/3} \rceil + 1)\frac{n^{0.5+0.1/3}}{0.9} \approx 10n^{0.53}$.

We used 48 ONEMAX instances with sizes $n = 20$, 30, 40,. . . , 490 and repeated CE 100 times for each instance. In each trial, the algorithm was allowed to only run at most $\lceil n^{0.53} \rceil$ iterations, but not $10n^{0.53}$ iterations suggested by Theorem 1. Note that this may reduce the chance to find the optimal solution in each trial. We stopped the algorithm immediately (before $\lceil n^{0.53} \rceil$ iterations) in each trial once the optimal solution was found.

Figure 1 (a) shows the relative frequency of trials in which the optimal solution was found within the 100 repetitions. For instances with sizes $n \ge 100$, this empirical success probability is well above the lower bound $1 - 2^{-n^{\epsilon/3}} \approx 1 - 2^{-n^{0.0333}}$ guaranteed by Theorem 1 a), although we actually run fewer than $10n^{0.53}$ iterations in each trial. The empirical probability equals 1 for the vast majority of instances with sizes $n \ge 200$. This means that our analysis is a little pessimistic, and still has room for further improvements of the probability lower bound and the upper bound of τ. Figure 1 (b) gives the average number of iterations needed to find the optimal solution from the 100 repetitions. It shows that the actual number of iterations is well below $n^{0.53}$. This further demonstrates the pessimism of our analysis of the bound of τ.

Finally, Figure 1 (c) shows the average CPU time needed for each instance. The CPU time should be proportional to the theoretical runtime $n^{0.53} \cdot N = n^{2.23}/15$. Figure 1 (c) also shows a quadratic function $an^{2.23} + bn + c$ fitted to the average CPU times for $n > 300$ with $a = 1.22 \cdot 10^{-5}$. The CPU times seem to be in good accordance with this function. This is another empirical evidence for the findings of Theorem 1 that the optimal solution for ONEMAX is found almost always with $O(n^{2.23})$ samples.



(a) Empirical probability (b) Actual time

Fig. 2: Empirical study of CE for LEADINGONES

2) Empirical study of LEADINGONES*:* For the empirical study of CE with the LEADINGONES problem, we used 18 LEADINGONES instances with sizes $n = 20, 30, 40, \ldots, 190$. Similarly, we started with the uniform distribution, and put $\rho = 0.9$. To test Theorem 2 a), we again use $\epsilon = 0.1$, and set $M = \lceil \frac{n^{2.1}}{500} \rceil, N = M \cdot \lceil n^{0.6} \rceil = \lceil n^{2.7} \rceil$. Theorem 2 a) then tells us that $\tau \le n$ with a probability at least $1 - e^{-n^{0.1}}$. To test Theorem 2 b), we take $\epsilon = 0.6, M = \lceil n^{\epsilon/2} \rceil = \lceil n^{0.3} \rceil, N = \lceil n^{1+\epsilon} \rceil = \lceil n^{1.6} \rceil, \delta_{\min} = \frac{1}{n}, \delta_{\max} = 1 - \frac{1}{n}$. Theorem 2 b) tells that in this case, $\tau \le n$ with a probability at least 1–$e^{-n^{0.6}}$. We also did 100 independent trials for each of the 18 instances in both of the two cases. To facilitate our discussion, we denote the results for CE with (6) by CElb.

Figure 2 (a) shows the relative frequency of trials in which the optimal solution was found within the 100 repetitions for both CE and CElb. It shows that the empirical success probability of CElb equals 1.0 for each instance, and therefore empirically validates the probability lower bound $1 - e^{-n^{0.6}}$ of Theorem 2 b). Note that $1 - e^{-n^{0.6}}$ is already 0.9976 when $n = 20$. Figure 2 (a) also shows empirical evidence for

Theorem 2 a), i.e., the empirical success probability of CE is well above the probability lower bound $1-e^{-n^{0.1}}$ for $n \geq 50$.

Figure 2 (b) shows the average CPU times of the 100 trials for each instance in the two cases. For $n \leq 140$, CE and CElb almost have the same CPU time. But as n becomes larger, the CPU time of CElb becomes considerably shorter than that of CE. Figure 2 (b) also uses two curves $0.000006n^{2.45}$ and $0.000006n^{2.65}$ to bound the actual CPU times. We see that the CPU time of CElb is bounded from above by $0.000006n^{2.45}$ and the CPU time of CE is bounded from above by $0.000006n^{2.65}$, which are well below the corresponding theoretical runtimes $n^{1.6} \cdot n = n^{2.6}$ and $n^{2.7} \cdot n = n^{3.7}$ of Theorem 2.

IV. Conclusion

We have shown that CE is very efficient for the two standard test problems OneMax and LeadingOnes. In particular, our results show that the general CE algorithm may outperform the more specific 1-ANT on OneMax, allowing for large samples in each iteration. This applies also in the case of more difficult fitness landscapes as in LeadingOnes, particularly if we bound the sampling probability away from 0 and 1 as it is done in 1-ANT and MMAS.

Though we obtained excellent runtime results on OneMax and LeadingOnes, the simple structure of these problems limits the scope of our results. In the future, we would therefore like to analyze CE on some standard problems in combinatorial optimization such as the minimum spanning tree problem [40] and the shortest path problem [26], or even some simple instances of $\mathcal{NP}$-hard problems as in [23] and [24].

Appendix

The proof of Theorem 1

The proof of Theorem 1 involves properties of independent Bernoulli trials with different success probabilities, and the update equation (5). We will inspect such Bernoulli trials more closely before starting the proof, and refer to [13] or [12] for necessary properties of the update equation.

Let $B_1, \dots, B_n$ be independent Bernoulli random variables with values 0 (failure) or 1 (success), and success probabilities $b_1, \dots, b_n$, respectively. Then,

$$B = \sum_{i=1}^{n} B_n \tag{9}$$

denotes the number of successes (ones) in the n independent trials. If $b_i \equiv p \in [0,1]$, then B obviously has the Binomial distribution.

Lemma 2 is the well-known Chernoff bound as it will be repeatedly used below. It states that the number B of successes is not likely to be much smaller or much larger than its expectation. See [41] for a proof.

Lemma 2 (Usual Chernoff bound)**.** *Let $B_1, \dots, B_n$ be independent Bernoulli random variables with success probabilities $b_1, \dots, b_n$, and $\mu = \sum_{i=1}^n b_i$. Then, for B as in* (9) *and $\delta \in (0,1)$*

a) $\mathbf{P}[B > (1+\delta)\mu] \leq 1 - e^{-\delta^2 \mu/3}$, *and*

b) $\mathbf{P}[B < (1-\delta)\mu] \leq 1 - e^{-\delta^2 \mu/2}$.

The proof of Theorem 1 will depend on the refined bound for B given in the following Lemma.

Lemma 3. *Let $B_1, \dots, B_n$ be n independent Bernoulli random variables with success probabilities $b_1, \dots, b_n$ respectively, $\mu = \sum_{i=1}^n b_i$ and B as in (9). Let $p > 0$ be a constant independent of n such that $b_i \geq p > 0$ for all $i = 1, 2, \dots, n$. Then for any $d \geq \frac{1}{p}$ independent of n we have for all $n \in \mathbb{N}$*

$$\mathbf{P}(B \geq \min\{\mu + d\sqrt{n - \lfloor \mu \rfloor},\ n\}) \geq \kappa > 0 \tag{10}$$

for some κ that may depend on d and p, but not on n.

Proof of Lemma 3. Note that whenever $p = 1$, the statement holds with $\kappa = 1$. Now, we assume $0 < p < 1$ and fix a constant $d \geq 1/p$. We first consider the homogeneous case $b_1 = \cdots = b_n = p > 0$ and prove the following stronger statement

$$\mathbf{P}(B \geq \min\{\mu + d \cdot \sqrt{n}, n\}) \geq \xi > 0, \tag{11}$$

where ξ is a constant depending only on p and d and not on n. As $p < 1$ and d is constant, there must be a $T \in \mathbb{N}$ such that for any $n \geq T$ $np + d \cdot \sqrt{n} < n$, and hence the minimum in (11) is attained for $\mu + d\sqrt{n}$. For this case we show that there exists $T' > T$ such that

$$\mathbf{P}(B \geq np + d \cdot \sqrt{n}) \geq \xi_1 > 0 \tag{12}$$

for a constant $\xi_1 > 0$ depending only on p, d. This follows directly from the Central Limit Theorem (CLT), which in our case states that

$$\mathbf{P}(B \geq np + d \cdot \sqrt{n}) = \mathbf{P}(\frac{B - np}{\sqrt{np(1-p)}} \geq \frac{d}{\sqrt{p(1-p)}})$$
$$\to 1 - \Phi(\frac{d}{\sqrt{p(1-p)}})$$

as $n \to \infty$. Hence, there exists a constant $\xi_1 > 0$ depending only on p, d and a $T' \geq T$ such that for any $n \geq T'$ (12) holds. For $n < T'$ we use $\xi_2 := p^{T'} > 0$ and

$$\mathbf{P}(B \geq \min\{\mu + d\sqrt{n}, n\}) \geq \mathbf{P}(B \geq n) = p^n \geq \xi_2 > 0,$$

then $\xi := \min\{\xi_1, \xi_2\}$ fulfills (11).

For the general, non-homogeneous case we compare $B_1, \dots, B_n$ with another sequence $Z_1, \dots, Z_n$ of independent Bernoulli random variables with success probabilities $z_1, \dots, z_n$. We again have $z_i \geq p$ and $\sum_{i=1}^n z_i = \sum_{i=1}^n b_i = \mu$, but the weight μ has been shifted as far as possible to the Z_i with smaller indices as follows. We define $m = \lfloor \frac{\mu - np}{1-p} \rfloor$, and let $Z_1, \dots, Z_m$ all get success probability 1 and $Z_{m+2}, \dots, Z_n$ get $z_i = p$, more precisely

$$z_i = \begin{cases} 1 & \text{for } i = 1, \dots, m \\ \mu - m - (n-m-1)p & \text{for } i = m+1 \\ p & \text{for } i = m+2, \dots, n \end{cases} .$$

Note that $z_{m+1} = p$ if $\frac{\mu - np}{1-p}$ is an integer, it guarantees $\sum_{i=1}^n z_i = \mu$. It is easy to check that $z_{m+1} \geq p$ and that $(z_1, \dots, z_n)$ majorizes $(b_1, \dots, b_n)$, i.e., $\sum_{i=1}^n z_i = \mu$ and

$$z_{[1]} \geq b_{[1]}, z_{[1]} + z_{[2]} \geq b_{[1]} + b_{[2]}, \dots, \sum_{i=1}^{n-1} z_{[i]} \geq \sum_{i=1}^{n-1} b_{[i]},$$

where the $z_{[i]}$'s and $b_{[i]}$'s are descending orderings of the z_i's and b_i's, respectively.

The distribution of the $Z_1, \dots, Z_n$ is easier to determine as the first m have succes probability 1. We have for any $r \geq m{+}1$

$$\begin{aligned}\mathbf{P}\Big(\sum_{i=1}^{n} Z_i \geq r\Big) &= \mathbf{P}\Big(Z_{m+1} + \sum_{i=1}^{m} Z_i \geq r - m\Big) \qquad (13)\\ &\geq \mathbf{P}\Big(\sum_{i=m+2}^{m} Z_i \geq r - m - 1,\ Z_{m+1} = 1\Big)\\ &= z_{m+1} \cdot \mathbf{P}\Big(\sum_{i=1}^{n-(m+1)} Z_{m+1+i} \geq r - (m+1)\Big)\\ &\geq p \cdot \mathbf{P}\Big(\sum_{i=1}^{n'} Z'_i \geq r - (m+1)\Big),\end{aligned}$$

where $n' = n - (m+1)$ and $Z'_1, \dots, Z'_{n'}$ are independent Bernoulli variables with constant success probability p. For $r = n$ we note for later reference that

$$\mathbf{P}\Big(\sum_{i=1}^{n'} Z'_i \geq n - (m+1)\Big) \ \geq\ pp^{n-(m+1)} = p^{n-m}. \qquad (14)$$

As $(z_1, \dots, z_n)$ majorizes $(b_1, \dots, b_n)$, we obtain from [42]

$$\mathbf{P}(B = n) \geq \mathbf{P}(\sum_{i=1}^{n} Z_i = n), \qquad (15)$$

and from [43]

$$\mathbf{P}\Big(B \geq \mu + c\Big) \geq \mathbf{P}\Big(\sum_{i=1}^{n} Z_i \geq \mu + c\Big) \qquad (16)$$

for any constant c with

$$c \geq 3 \quad \text{and} \quad \mu + c < n. \qquad (17)$$

Let us first assume that (17) holds for $c = d\ \sqrt{n - \lfloor\mu\rfloor}$. Then the minimum in (10) is again attained for the first term, and $d\sqrt{n - \lfloor u\rfloor} < n - \mu$. As we assumed $d \geq 1/p$ we also have

$$\begin{aligned}\lfloor\mu\rfloor - m - 1 &\geq \mu - 1 - \frac{\mu - np}{1-p} - 1 = \frac{p(n-\mu)}{1-p} - 2\\ &\geq \frac{p \cdot d^2}{1-p} - 2 \geq \frac{1}{p(1-p)} - 2 \geq 4 - 2 = 2 > 0,\end{aligned}$$

hence $\lfloor\mu\rfloor > m + 1$ and from (16) and (13) we get

$$\begin{aligned}\mathbf{P}\Big(B \geq \mu + d\sqrt{n - \lfloor\mu\rfloor})\Big) &\geq \mathbf{P}\Big(\sum_{i=1}^{n} Z_i \geq \mu + d\sqrt{n - \lfloor\mu\rfloor}\Big)\\ &\geq \mathbf{P}\Big(\sum_{i=1}^{n} Z_i \geq \mu + d\sqrt{n - (m+1)}\Big)\\ &\geq p \cdot \mathbf{P}\Big(\sum_{i=1}^{n'} Z'_i \geq \mu - (m+1) + d\sqrt{n - (m+1)}\Big)\\ &\geq p \cdot \mathbf{P}\Big(\sum_{i=1}^{n'} Z'_i \geq \big(n - (m+1)\big)p + d\sqrt{n - (m+1)}\Big)\\ &\geq p \cdot \xi = \kappa_1 > 0.\end{aligned}$$

Here we used $\mu - (m+1) = \sum_{i=m+2}^{n} z_i = (n - (m+1))p$ and applied the homogeneous result (11) to $Z'_1, \dots, Z'_{n'}$ with $n' = n - (m+1)$.

Now we have to check the case where (17) does not hold for $c = d\ \sqrt{n - \lfloor\mu\rfloor}$. We use

$$\mathbf{P}\big(B \geq \min\{\mu + d\sqrt{n - \lfloor\mu\rfloor},\ n\}\big) \geq \mathbf{P}\big(B \geq n\big)$$

and first assume that $c = d\ \sqrt{n - \lfloor\mu\rfloor} < 3$. Then $\lfloor\mu\rfloor > n - \frac{9}{d^2}$ and hence $n - m \leq \frac{9}{d^2(1-p)} + 1$. From (15) and (14) we obtain for all n

$$\mathbf{P}\big(B \geq n\big) \geq p^{n-m} \geq p^{\frac{9}{d^2(1-p)}+1} = \kappa_2 > 0.$$

Finally, if the second condition in (17) is violated, then $\mu \geq n - d\sqrt{n - \lfloor\mu\rfloor}$ leads to $\mu \geq n - \frac{d}{2}(d + \sqrt{4 + d^2})$. Therefore, $n - m \leq \frac{d(d+\sqrt{4+d^2})}{2(1-p)} + 1$ and

$$\mathbf{P}\big(B \geq n\big) \geq p^{\frac{d(d+\sqrt{4+d^2})}{2(1-p)}+1} = \kappa_3 > 0.$$

Now $\kappa := \min\{\kappa_1, \kappa_2, \kappa_3\} > 0$ fufills (10) . □

Lemma 3 applies to every iteration t and says that, if the entries of model $\mathbf{\Pi}_t$ are uniformly bounded away from 0 by a constant not depending on n, then in each draw of iteration t the probability of generating a solution with fitness $\sum_{i=1}^{n} p_{t,i} + \Theta(\sqrt{n - \mu})$ is relatively large, i.e., in $\Omega(1)$. And if $\sqrt{n - \mu} \in O(1)$, the probability of generating the optimal solution in a draw of iteration t is even $\Omega(1)$. Therefore, if Algorithm 1 starts with a positive initial model, it reaches the optimal solution very quickly. In the proof of Theorem 1 we will formally assert this.

Lemma 4 shows that if the elite sample size M is not too small, then the probability of generating '1' at any position will significantly deteriorate in the reinforcement learning step (5) of CE.

Lemma 4. *Consider the* ONEMAX *problem. Fix an iteration $t \in \mathbb{N}$, and a corresponding model $\mathbf{\Pi}_t = (p_{t,1}, \dots, p_{t,n})$ with each $p_{t,i} \in \Omega(1)$. For any $\delta \in (0, \rho)$ we have that*

$$\mathbf{P}\big(p_{t+1,i} \geq p_{t,i}(1-\delta), i = 1\dots, n\big) \geq 1 - ne^{-\frac{\delta^2}{2\rho^2}\bar{p}M},$$

where $\bar{p} = \min\{p_{t,i} : i = 1, \dots, n\}$.

Proof of Lemma 4. Note that if for each $i = 1, \dots, n$,

$$\mathbf{P}[p_{t+1,n} \geq p_{t,n}(1-\delta)] \geq 1 - e^{-\frac{\delta^2 \bar{p}M}{2\rho^2}}$$

holds, then the statement follows. Notice that

$$w_{t,i} \geq p_{t,i}(1 - \frac{\delta}{\rho}) \iff p_{t+1,n} \geq p_{t,n}(1-\delta),$$

where $w_{t,i}$ is defined in (2).

Now, we arbitrarily fix a position $i \in \{1, \dots, n\}$. Let $\mathcal{M} \subseteq \mathcal{N} := \{1, 2, \dots, N\}$ be indices of the M elite solutions in the sample $\mathbf{X}_t = (X_1^{(1)}, \dots, X_t^{(N)})$, i.e., for each $j \in \mathcal{N}$, $j \in \mathcal{M}$ iff $X_t^{(j)} = X_t^{[m]}$ for some $m \in \{1, \dots, M\}$. Obviously, $\mathcal{M}$ is a random set. Let $\mathfrak{M}$ be the collection of subsets of $\mathcal{N}$ having size M and put $q := Mp_{t,i}(1 - \frac{\delta}{\rho})$. Note that if

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2667713, IEEE Transactions on Evolutionary Computation

$\mathcal{M} = \mathbb{M} \in \mathfrak{M}$, then $w_{t,i} \geq p_{t,i}(1-\frac{\delta}{\rho})$ iff $\sum_{j\in\mathbb{M}} X_t^{(j)}(i) \geq q$. Therefore, we obtain

$$\mathbf{P}\Big[w_{t,i} \geq p_{t,i}(1-\frac{\delta}{\rho})\Big] = \sum_{\mathbb{M}\in\mathfrak{M}} \mathbf{P}\Big[w_{t,i} \geq p_{t,i}(1-\frac{\delta}{\rho}), \mathcal{M}=\mathbb{M}\Big]$$
$$= \sum_{\mathbb{M}\in\mathfrak{M}} \mathbf{P}\Big[\sum_{j\in\mathbb{M}} X_t^{(j)}(i) \geq q, \mathcal{M}=\mathbb{M}\Big]. \tag{18}$$

For an *arbitrarily fixed* set $\mathbb{M} \in \mathfrak{M}$,

$$\mathbf{P}\Big[\sum_{j\in\mathbb{M}} X_t^{(j)}(i) \geq q, \mathcal{M} = \mathbb{M}\Big] \tag{19}$$
$$= \mathbf{P}\Big[\sum_{j\in\mathbb{M}} X_t^{(j)}(i) \geq q\Big] \cdot \mathbf{P}\Big[\mathcal{M} = \mathbb{M}\Big| \sum_{j\in\mathbb{M}} X_t^{(j)}(i) \geq q\Big].$$

Now, we want to show that for the fixed $\mathbb{M} \in \mathfrak{M}$, conditioned on the event $\sum_{j\in\mathbb{M}} X_t^{(j)}(i) \geq q$, probability of the event that $\mathcal{M} = \mathbb{M}$ (i.e. the elite part in $\mathbf{X}_t$ is $\{X_t^{(j)} : j \in \mathbb{M}\}$), is increased in iteration t.

For the fixed $\mathbb{M} \in \mathfrak{M}$, let $v_* := \max\{f(X_t^{(k)}) : k \in \mathcal{N} - \mathbb{M}\}$ be the maximum fitness value of the $N - M$ solutions with indices in $\mathcal{N} - \mathbb{M}$, and $\zeta := \min\{k : k \in \mathcal{N} - \mathbb{M}, f(X_t^{(k)}) = v_*\}$ be the index of the first solution in these $N-M$ solutions having fitness value v_*. Obviously, for any fixed $\mathbb{M} \in \mathfrak{M}$, $\sum_{j\in\mathbb{M}} X_t^{(j)}(i)$ is independent of v_* and ζ, since they concern two mutually exclusive parts of the N independent solutions contained in $\mathbf{X}_t$. For any $k \in \mathcal{N} - \mathbb{M}$ and fitness value v, conditioned on $\zeta = k$ and $v_* = v$, the event $\mathcal{M} = \mathbb{M}$ is equivalent to the random event $\mathcal{E}$ that $f(X_t^{(j)}) > v$ if $j > k$, and $f(X_t^{(j)}) \geq v$ if $j < k$, for any $j \in \mathbb{M}$, since we sort the N solutions under the convention that if two solutions have the same fitness value, we consider the solution generated earlier as the better one. Then, we further obtain

$$\mathbf{P}\Big[\mathcal{M} = \mathbb{M}\Big| \sum_{j\in\mathbb{M}} X_t^{(j)}(i) \geq q\Big]$$
$$= \sum_{v=0}^{n} \sum_{k\in\mathcal{N}-\mathbb{M}} \mathbf{P}\Big[\mathcal{M}=\mathbb{M}\Big|\zeta=k, v_*=v, \sum_{j\in\mathbb{M}} X_t^{(j)}(i) \geq q\Big]$$
$$\cdot \mathbf{P}\Big[\zeta=k, v_* = v\Big| \sum_{j\in\mathbb{M}} X_t^{(j)}(i) \geq q\Big]$$
$$= \sum_{v=0}^{n} \sum_{k\in\mathcal{N}-\mathbb{M}} \mathbf{P}\Big[\mathcal{E}\Big|\zeta=k, v_*=v, \sum_{j\in\mathbb{M}} X_t^{(j)}(i) \geq q\Big]$$
$$\cdot \mathbf{P}\Big[v_* = v\Big] \cdot \mathbf{P}\Big[\zeta = k\Big|v_* = v\Big]. \tag{20}$$

Now, a crucial step in our proof is to show that (20) is actually not smaller than $\mathbf{P}[\mathcal{M} = \mathbb{M}]$. To achieve this, we show first for any $c \leq M - 1$ that

$$\mathbf{P}\Big[\mathcal{E}\Big|\zeta=k, v_*=v, \sum_{j\in\mathbb{M}} X_t^{(j)}(i) = c\Big] \tag{21}$$
$$\leq \mathbf{P}\Big[\mathcal{E}\Big|\zeta=k, v_*=v, \sum_{j\in\mathbb{M}} X_t^{(j)}(i) = c+1\Big].$$

Given $v_* = v, \zeta = k$ and $\sum_{j\in\mathbb{M}} X_t^{(j)}(i) = c$ for some constant $c \in \{0, 1, \ldots, M\}$, the probability of $\mathcal{E}$ equals 0 if $v = n$ and $k < j$ for some $j \in \mathbb{M}$ since $k \notin \mathbb{M}$ and the sorting convention of Algorithm 1. Now, we consider the probability of $\mathcal{E}$ when $v_* = v < n$. Let $\alpha := \{j \in \mathbb{M} : j < k\}$ for the given k. Let $Y = (y_1, \ldots, y_n)$ be a random solution drawn from $\mathbf{\Pi}_t$, and put $\kappa := \mathbf{P}[\sum_{u\neq i} y_u \geq v - 1], \theta := \mathbf{P}[\sum_{u\neq i} y_u \geq v]$, and $\lambda := \mathbf{P}[\sum_{u\neq i} y_u \geq v+1]$. Obviously, $\kappa \geq \theta \geq \lambda$. Conditioned on $v_* = v < n, \zeta = k$, $\mathcal{E}$ holds iff for the α many $j \in \mathbb{M}$ with $j < k$ $f(X_t^{(j)}) \geq v$ and for the remaining $M - \alpha$ many $j \in \mathbb{M}$ $f(X_t^{(j)}) > v$. For these $j \in \mathbb{M}$ with $j < k$, $\mathbf{P}[j \in \mathcal{M}|X_t^{(j)}(i) = 1] = \kappa$, $\mathbf{P}[j \in \mathcal{M}|X_t^{(j)}(i) = 0] = \theta$. For these $j \in \mathbb{M}$ with $j > k$, $\mathbf{P}[j \in \mathcal{M}|X_t^{(j)}(i) = 1] = \theta$, $\mathbf{P}[j \in \mathcal{M}|X_t^{(j)}(i) = 0] = \lambda$. Therefore, we obtain for $v < n$ that

$$\mathbf{P}\Big[\mathcal{E}\Big|\zeta=k, v_*=v, \sum_{j\in\mathbb{M}} X_t^{(j)}(i) = c\Big] \tag{22}$$
$$= \sum_{c_1=\max\{0,c+\alpha-M\}}^{\min\{\alpha,c\}} \frac{\binom{\alpha}{c_1}\binom{M-\alpha}{c-c_1}}{\binom{M}{c}} \kappa^{c_1} \theta^{\alpha+c-2c_1} \lambda^{M-\alpha-c+c_1}$$
$$:= L(c, M, \alpha),$$

Herein we actually only need to consider arrangement of the c many ones in rows $\in \mathbb{M}$ in column i, and if exactly c_1 many ones are in rows $\in \{j \in \mathbb{M} : j < k\}$, then $c - c_1$ many ones have to be in rows $\in \{j \in \mathbb{M} : j > k\}$. Therefore the second equation in (22) follows, since each arrangement of c many ones has the probability $p_{t,i}^c(1-p_{t,i})^{M-c}$. Now, we inspect $L(c+1, M, \alpha)$ more closely for some c with $c + 1 \leq M$. We assume, w.l.o.g., that $c < \min\{\alpha, M - \alpha\}$. We can first put *one* of the $c+1$ ones in a row $\in \mathbb{M}$ in column i, then arrange the remaining c many ones in the remaining $M-1$ rows $\in \mathbb{M}$, this results in

$$L(c+1,M,\alpha) = \sum_{c_1=0}^{c} \frac{\binom{\alpha}{1}\kappa\binom{\alpha-1}{c_1}\binom{M-\alpha}{c-c_1}\kappa^{c_1}\theta^{\alpha+c-2c_1-1}\lambda^{M-\alpha-c+c_1}}{(c+1)\binom{M}{c+1}}$$
$$+ \sum_{c_1=0}^{c} \frac{\binom{M-\alpha}{1}\theta\binom{\alpha}{c_1}\binom{M-\alpha-1}{c-c_1}\kappa^{c_1}\theta^{\alpha+c-2c_1}\lambda^{M-\alpha-c+c_1-1}}{(c+1)\binom{M}{c+1}}$$
$$\geq \min\{\frac{\kappa}{\theta}, \frac{\theta}{\lambda}\} \sum_{c_1=0}^{c} \frac{\binom{\alpha}{c_1}\binom{M-\alpha}{c-c_1}\kappa^{c_1}\theta^{\alpha+c-2c_1}\lambda^{M-\alpha-c+c_1}}{\binom{M}{c}}$$
$$\geq L(c, M, \alpha),$$

where the $c+1$ in the denominator denotes the number of repetitions in our calculation for every arrangement of the $c+1$ many ones. Therefore, (21) is fulfilled.

With (21), we can easily finish the proof. Note that

$$\mathbf{P}\Big[\mathcal{E}\Big|\zeta=k, v_*=v, \sum_{j\in\mathbb{M}} X_t^{(j)}(i) \geq q\Big] \tag{23}$$
$$= \sum_{c=q}^{n} \mathbf{P}\Big[\mathcal{E}\Big|\zeta=k, v_*=v, \sum_{j\in\mathbb{M}} X_t^{(j)}(i) = c\Big]$$
$$\cdot \mathbf{P}\Big[\sum_{j\in\mathbb{M}} X_t^{(j)}(i) = c\Big| \sum_{j\in\mathbb{M}} X_t^{(j)}(i) \geq q\Big].$$

$$\mathbf{P}\Big[\mathcal{E}\Big|\zeta=k, v_* = v\Big] = \sum_{c=0}^{n} \mathbf{P}\Big[\mathcal{E}\Big|\zeta=k, v_*=v, \sum_{j\in\mathbb{M}} X_t^{(j)}(i) = c\Big]$$
$$\cdot \mathbf{P}\Big[\sum_{j\in\mathbb{M}} X_t^{(j)}(i) = c\Big]. \tag{24}$$

Therefore, we obtain from (21) that

$$\mathbf{P}\Big[\mathcal{E}\Big|\zeta=k, v_*=v\Big] \leq \mathbf{P}\Big[\mathcal{E}\Big|\zeta=k, v_*=v, \sum_{j\in\mathbb{M}} X_t^{(j)}(i)\geq q\Big], \quad (25)$$

since (23) averages the higher values and (24) calculates the total average.

By (25) and (20), we have that

$$\begin{aligned}&\mathbf{P}\Big[\mathcal{M}=\mathbb{M}\Big|\sum_{j\in\mathbb{M}} X_t^{(j)}(i)\geq q\Big]\\ &\geq \sum_{v=0}^{n}\sum_{k\in\mathcal{N}-\mathbb{M}} \mathbf{P}\Big[\mathcal{M}=\mathbb{M}\Big|\zeta=k, v_*=v\Big] \quad (26)\\ &\quad\cdot \mathbf{P}\Big[v_*=v\Big]\cdot\mathbf{P}\Big[\zeta=k\Big|v_*=v\Big]=\mathbf{P}[\mathcal{M}=\mathbb{M}].\end{aligned}$$

By (19) and (26), we have for a fixed $\mathbb{M}\in\mathfrak{M}$ that

$$\begin{aligned}&\mathbf{P}\Big[\sum_{j\in\mathbb{M}} X_t^{(j)}(i)\geq q, \mathcal{M}=\mathbb{M}\Big]\\ &\geq \mathbf{P}\Big[\sum_{j\in\mathbb{M}} X_t^{(j)}(i)\geq Mp_{t,i}(1-\frac{\delta}{\rho})\Big]\mathbf{P}[\mathcal{M}=\mathbb{M}] \quad (27)\\ &\geq(1-e^{-\frac{\delta^2 p_{t,i}M}{2\rho^2}})\mathbf{P}[\mathcal{M}=\mathbb{M}]\geq(1-e^{-\frac{\delta^2\bar{p}M}{2\rho^2}})\mathbf{P}[\mathcal{M}=\mathbb{M}],\end{aligned}$$

where we apply the usual Chernoff bound. Combining (18) and (27) proves the statement. □

With the help of Lemma 3 and Lemma 4, we are now ready to prove Theorem 1. The proof shows that for each iteration $t\geq 1$, $f(X_t^{[M]})-f(X_{t-1}^{[M]})\in\Omega(\sqrt{n-f(X_{t-1}^{[M]})})$ with an overwhelming probability if we set $M=\Omega(N^{1+\epsilon})$, $N=\omega(M)$ and use a strictly positive $\mathbf{\Pi}_0$ (i.e. each $p_{0,i}\in\Omega(1)$). Lemma 3 asserts that if $\mathbf{\Pi}_{t-1}$ is strictly positive, this happens with an overwheming probability. Lemma 4 asserts that if $\mathbf{\Pi}_0$ is strictly positive, $\mathbf{\Pi}_t$ is still positive for any $t\in O(\frac{n^{0.5+\epsilon/3}}{\rho})$.

Proof of Theorem 1: Here, we assume, w.l.o.g., that $p_{0,i}=1/2$ for all $i=1,2,3,\ldots,n$. For any other initial model $\mathbf{\Pi}_0$ with $p_{0,i}\in\Omega(1)$ the proof is the same.

Let $\eta=\epsilon/3\in(0,\frac{1}{3})$ and $K_\eta:=\lceil\frac{1}{4\eta}\rceil$. Obviously, $1-2K_\eta\cdot\eta<\frac{1}{2}$. We shall consider the probability of the event that the optimal solution is sampled within $(\lceil\frac{1}{4\eta}\rceil+1)\frac{n^{1/2+\eta}}{\rho}$ iterations. We divide the whole optimization procedure into $K_\eta+1$ stages. We inspect for each stage l with $1\leq l\leq K_\eta$, the probability that the threshold value $f(X_t^{[M]})$ eventually exceeds $n-n^{1-2l\eta}$ by increasing its value by at least $\rho n^{1/2-l\eta}$ per iteration. In the last stage $l=K_{\eta+1}$, we investigate the probability of the event that the threshold value eventually reaches n, ie., the optimal solution is found in this stage, by increasing its value by at least ρ per iteration.

We now analyze the number of iterations needed for achieving the corresponding task in each stage. The first stage takes at most $\tau_1:=\frac{\frac{n}{2}-n^{1-2\eta}-n^{1/2-\eta}}{\rho n^{1/2-\eta}}\leq\frac{n^{\frac{1}{2}+\eta}}{2\rho}$ iterations provided that the threshold value increases by at least $\rho n^{1/2-\eta}$ in each iteration, and $f(X_0^{[M]})\geq\frac{n}{2}+n^{1/2-\eta}$ at the beginning. Each stage l, $2\leq l\leq K_\eta$, takes at most $\tau_l:=\frac{n^{1-2(l-1)\eta}-n^{1-2l\eta}}{\rho n^{\frac{1}{2}-l\eta}}\leq\frac{1}{\rho}n^{\frac{1}{2}-(l-2)\eta}<\frac{1}{\rho}n^{1/2}$ iterations if the threshold value increases at least by $\rho n^{1/2-l\eta}$ per iteration in that stage. The last stage requires at most $\tau_{K_\eta+1}:=\frac{1}{\rho}n^{1-2K_\eta\eta}<\frac{1}{\rho}n^{1/2}$ iterations. Hence, if the threshold value increases as described above, the optimal solution is obtained in at most $T:=\sum_{l=1}^{K_\eta+1}\tau_l\leq(K_\eta+1)\frac{n^{1/2+\eta}}{\rho}=(\lceil\frac{1}{4\eta}\rceil+1)\frac{n^{1/2+\eta}}{\rho}$ iterations, as is stated in Theorem 1.

The following shows that the assumptions made for the derivations of the τ_l above hold with the overwhelming probability given in Theorem 1. This proves the Theorem.

To this end, we consider some particular random events $\mathcal{E}_t$ in the first T iterations. Let $T_l:=\sum_{k=1}^{l}\tau_l$ for $1\leq l\leq K_\eta+1$, with $T_0:=0$ and $T=T_{K_\eta+1}$. For each $0\leq t<\tau_1$, we define the random event $\mathcal{E}_t$ by requiring A$_1$)-A$_3$) below simultaneously:

A$_1$) $f(X_t^{[k]})\geq\frac{n}{2}+t\rho n^{1/2-\eta}+n^{1/2-\eta}$ for each $k=1,\ldots,M$,
A$_2$) $\sum_{i=1}^{n}p_{t,i}\geq\frac{n}{2}+t\rho n^{1/2-\eta}$,
A$_3$) $p_{t+1,i}\geq\frac{1}{2}(1-\frac{1}{T})^t>\frac{1}{2e}$ for each $i=1,\ldots,n$.

For each following stage $1<l\leq K_\eta$ and each iteration $T_{l-1}\leq t<T_l$ in that stage, the random event $\mathcal{E}_t$ is the event that B$_1$)-B$_3$) below happen simultaneously:

B$_1$) $f(X_t^{[k]})\geq n-n^{1-2(l-1)\eta}+\rho(t-T_{l-1})n^{1/2-l\eta}+n^{1/2-l\eta}$ for each $k=1,\ldots,M$,
B$_2$) $\sum_{i=1}^{n}p_{t,i}\geq n-n^{1-2(l-1)\eta}+\rho(t-T_{l-1})n^{1/2-l\eta}$,
B$_3$) $p_{t+1,i}\geq\frac{1}{2}(1-\frac{1}{T})^t>\frac{1}{2e}$ for each $i=1,\ldots,n$.

For the last stage $l=K_\eta+1$ and each iteration $T_{K_\eta}\leq t<T$ the random event $\mathcal{E}_t$ is defined by C$_1$)-C$_2$) taking place simultaneously:

C$_1$) $f(X_t^{[k]})\geq n-n^{1-2K_\eta\eta}+\rho(t-T_{K_\eta})+1$ for each $k=1,\ldots,M$,
C$_2$) $\sum_{i=1}^{n}p_{t,i}\geq n-n^{1-2K_\eta\eta}+\rho(t-T_{K_\eta})$,
C$_3$) $p_{t+1,i}\geq\frac{1}{2}(1-\frac{1}{T})^t>\frac{1}{2e}$ for each $i=1,\ldots,n$.

Obviously the random event $\mathcal{E}:=\bigcap_{l=1}^{K_\eta+1}\mathcal{E}_l^\diamond$ with $\mathcal{E}_l^\diamond:=\bigcap_{t=T_{l-1}}^{T_l}\mathcal{E}_j$ implies that the total number of iterations required to reach the optimal solution is bounded from above by T. We now bound the probability for the event $\mathcal{E}$ from below. We will show that $\mathbf{P}(\mathcal{E}_1^\diamond)\geq 1-e^{-\Omega(n^\eta)}$ and $\mathbf{P}[\mathcal{E}_{l+1}^\diamond\mid\bigcap_{j\leq l}\mathcal{E}_j^\diamond]\geq 1-e^{-\Omega(n^\eta)}$ for every $l=1.\ldots,K_\eta$. This implies $\mathbf{P}[\mathcal{E}]\geq(1-e^{-\Omega(n^\eta)})^{K_\eta+1}=1-e^{-\Omega(n^\eta)}$ which is overwhelmingly large with growing n.

We now prove for the first stage $l=1$ that $\mathbf{P}[\mathcal{E}_1^\diamond]\geq 1-e^{-\Omega(n^\eta)}$. When $t=0$, A$_2$) holds for sure. By Lemma 3, we know that the probability that a random solution contains at least $\frac{n}{2}+n^{1/2-\eta}$ many ones is bounded from below by some *constant* $\kappa>0$. When we take $p:=\frac{1}{2e}$ and $d:=2\sqrt{2}e$, then $\mu=\sum_{i=1}^{n}p_{0,i}=\frac{n}{2}$ and $d\sqrt{n-\lfloor\mu\rfloor}>n^{1/2-\eta}$ in this case. By the usual Chernoff bound, the probability that the first sample $\mathbf{X}_0$ contains at least M many solutions with at least $n/2+n^{1/2-\eta}$ many ones is bounded from below by $1-e^{-\frac{(1-\frac{M}{\kappa N})^2\Omega(N)}{2}}=1-e^{-\Omega(N)}$, since $N\in\omega(M)$ and $M=\Omega(n^{1+3\eta})$. This shows that A$_1$) happens with a probability at least $1-e^{-\Omega(N)}$.

We now inspect the probability of A$_3$). By Lemma 4, $\mathbf{P}\Big(p_{1,i}\geq\frac{1}{2}(1-\frac{1}{T}),\forall i=1,\ldots,n\Big)\geq 1-e^{-\Omega(n^\eta)}$ since $\rho^2T^2=O(n^{1+2\eta})$, $M=\Omega(n^{1+3\eta})$. Altogether, we have that $\mathbf{P}(\mathcal{E}_0)\geq 1-e^{-\Omega(n^\eta)}-e^{-\Omega(N)}=1-e^{-\Omega(n^\eta)}$.

Now, conditioned on $\mathcal{E}_0, \mathcal{E}_1, \dots, \mathcal{E}_j$ for any $j < \tau_1$, we inspect the probability of $\mathcal{E}_{j+1}$. Conditioned on $\mathcal{E}_j$, we know that

1) $f(X_j^{[k]}) \geq \frac{n}{2} + j\rho n^{1/2-\eta} + n^{1/2-\eta}$ for each $k = 1, \dots, M$;
2) $\sum_{i=1}^n p_{j,i} \geq \frac{n}{2} + j\rho n^{1/2-\eta}$;
3) $p_{j+1,i} \geq \frac{1}{2}(1-\frac{1}{T})^j > \frac{1}{2e}$ for each $i = 1, \dots, n$.

Conditioned on 1) and 2), A$_2$) holds at iteration $t = j+1$ for sure. With the help of A$_2$) and 3), we now bound the probability for A$_1$) from below at iteration $t = j+1$.

By A$_2$) and 3), we have

◇ $\sum_{i=1}^n p_{t,i} \geq \frac{n}{2} + t\rho n^{1/2-\eta}$;
◇ $p_{t,i} \geq \frac{1}{2}(1-\frac{1}{T})^{t-1} \geq \frac{1}{2e}$ for all $i = 1, \dots, n$.

By Lemma 3, the probability of generating a solution with fitness at least $C := \min\{\sum_{i=1}^n p_{t,i} + d\sqrt{n - \lfloor \sum_{i=1}^n p_{t,i} \rfloor},\ \ n\}$ in each of the N draws is bounded from below by the constant $\kappa > 0$, where again we take $p := \frac{1}{2e}$ and $d = 2\sqrt{2}e$. Note that $n \geq n - n^{1-2\eta} \geq \frac{n}{2} + t\rho n^{1/2-\eta} + n^{1/2-\eta}$ if $t \leq \tau_1$. If $\sum_{i=1}^n p_{t,i} \geq n - n^{1-2\eta}$, then obviously $C \geq \frac{n}{2} + t\rho n^{1/2-\eta} + n^{1/2-\eta}$. If $\sum_{i=1}^n p_{t,i} < n - n^{1-2\eta}$, then $\sqrt{n - \lfloor \sum_{i=1}^n p_{t,i} \rfloor} > n^{1/2-\eta}$, and by A$_2$) it holds again that $C \geq \frac{n}{2} + t\rho n^{1/2-\eta} + n^{1/2-\eta}$. Therefore, in each of the N draws in iteration t, the probability of generating a solution with fitness at least $\frac{n}{2} + t\rho n^{1/2-\eta} + n^{1/2-\eta}$ is again bounded from below by the constant $\kappa > 0$. Then, by the Chernoff bound, we obtain that $\mathcal{E}_t$ conditioned on $\mathcal{E}_0, \mathcal{E}_1, \dots, \mathcal{E}_j$ for any $j < \tau_1$ fulfills A$_1$) in iteration $t = j+1$ with probability at least $1 - e^{-\frac{(1-\frac{M}{\kappa N})^2 \Omega(N)}{2}} = 1 - e^{-\Omega(N)}$.

Again by Lemma 4, one can show that A$_3$) holds in iteration t again with a probability at least $1 - e^{-\Omega(n^\eta)}$, since $p_{t,i} \geq \frac{1}{2e}$ for all $i = 1, 2, \dots, n$ and every $t \leq T$.

This means that $\mathbf{P}[\mathcal{E}_{j+1} \mid \bigcap_{u=0}^j \mathcal{E}_u] \geq 1 - e^{-\Omega(n^\eta)}$ for any $j < \tau_1$. As a result, we obtain $\mathbf{P}[\mathcal{E}_1^\diamond] \geq 1 - e^{-\Omega(n^\eta)}$. Similarly, one can show that for any $l = 1, 2, \dots, K_\eta$, $\mathbf{P}[\mathcal{E}_{l+1}^\diamond \mid \bigcap_{j=1}^l \mathcal{E}_j^\diamond] \geq 1 - e^{-\Omega(n^\eta)}$ by observing the following facts,

◇ $n - n^{1-2l\eta} + \rho(t - T_l)n^{1/2-(l+1)\eta} + n^{1/2-(l+1)\eta} \leq \sum_{i=1}^n p_{t,i} + d\sqrt{n - \lfloor \sum_{i=1}^n p_{t,i} \rfloor}$ for any t in stage $l+1 \leq K_\eta$,
◇ $n - n^{1-2K_\eta \eta} + \rho(t - T_{K_\eta}) + 1 \leq \min\{\sum_{i=1}^n p_{t,i} + d\sqrt{n - \lfloor \sum_{i=1}^n p_{t,i} \rfloor},\ n\}$ for any t in stage $K_\eta + 1$,

where again $p = \frac{1}{2e}$ and $d = 2\sqrt{2}e$. ■

The proof of Theorem 2

Proof of Theorem 2: As in the proof of Theorem 1, we inspect the probability that the number of leading ones in each elite solution increases by a certain amount per iteration. However, this amount is at most $O(1)$ for a polynomial sample size N and a not very large sampling probability since simultaneously generating $\omega(1)$ many leading ones may have an exponentially small probability. Therefore, the total number of iterations to reach the optimal solution with an overwhelming probability needs to be at least $\Omega(n)$.

The proof of a) is very similar to the one in [13], readers may want to refer to [13].

a) We prove the statement by showing that the random event that the threshold value $f(X_t^{[M]})$ increases at least by 1 per iteration, has an overwhelming probability in our setting. To this end, we define a particular random event $\mathcal{E}_t$ as the intersection of the following three events for every iteration $t \in \{0, 1, 2, \dots, n-1\}$:

1) $\sum_{j=1}^N \prod_{i=1}^{t+1} X_t^{(j)}(i) \geq M$;
2) $p_{t+1,i} \geq 1 - (1 - p_{i-1,i})(1-\rho)^{t+2-i}$ for each $i = 1, \dots, t+1$;
3) $p_{t+1,i} \geq p_{0,i}(1-\frac{1}{n})^{t+1}$ for all $i = t+2, \dots, n$.

Obviously, the event $\mathcal{E} := \bigcap_{t=0}^{n-1} \mathcal{E}_t$ implies that the total number of iterations needed for Algorithm 1 to reach the optimal solution is $O(n)$. The remainder of the proof investigates the probability of this event $\mathcal{E}$.

We show that for each iteration $t = 0, 1, \dots, n-1$,

$$\mathbf{P}[\mathcal{E}_t \mid \bigcap_{l=0}^{t-1} \mathcal{E}_l] \geq 1 - e^{-\Omega(\frac{M}{\rho^2 n^2})} \tag{28}$$

with the convention that $\mathbf{P}[\mathcal{E}_0 \mid \bigcap_{l=0}^{-1} \mathcal{E}_l] := \mathbf{P}[\mathcal{E}_0]$. This implies directly statement a). We prove (28) only for the case that $t \geq 1$. The same argument carries over to the case $t = 0$.

Note that conditioned on $\bigcap_{l=0}^{t-1} \mathcal{E}_l$, we have for sure that

$$p_{t,i} \geq \begin{cases} 1 - (1 - p_{i-1,i})(1-\rho)^{t+1-i} & \text{if } i = 1, \dots, t; \\ p_{0,i}(1-\frac{1}{n})^t & \text{if } i = t+1, \dots, n. \end{cases}$$

Then, we can conclude that

$$\begin{aligned} \prod_{i=1}^{t+1} p_{t,i} &\geq p_{0,t+1}(1-\frac{1}{n})^t \prod_{i=1}^{t} (1 - (1-\rho)^{t+1-i}) \\ &\geq \frac{p_{\min}}{e} \prod_{u=1}^{\infty} (1 - (1-\rho)^u), \end{aligned}$$

which is in $\Omega(1)$ if $\rho \in \Omega(1)$ and if the first model $\mathbf{\Pi}_0$ is strictly positive, where $p_{\min} := \min \mathbf{\Pi}_0 = \min\{p_{0,1}, \dots, p_{0,n}\}$. Then we obtain from the Chernoff bound

$$\begin{aligned} &\mathbf{P}\left[\sum_{j=1}^{N} \prod_{i=1}^{t+1} X_t^{(j)}(i) \geq M \mid \bigcap_{l=1}^{t-1} \mathcal{E}_l\right] \\ &\qquad \geq 1 - e^{-(1-\frac{M}{\Omega(1)N})^2 \frac{N}{2}} = 1 - e^{-\Omega(N)}. \end{aligned} \tag{29}$$

This means that, conditioned on $\bigcap_{l=0}^{t-1} \mathcal{E}_l$, 1) holds with an overwhelming probability of $1 - e^{-\Omega(N)}$. Note that, conditioned on $\bigcap_{l=0}^{t-1} \mathcal{E}_l$ and 1), 2) holds for sure at time t. We now investigate the probability for 3) at time t, conditioned on 1) and $\bigcap_{l=0}^{t-1} \mathcal{E}_l$.

We define a random variable $Y_1 := \sum_{j=0}^N \prod_{i=1}^{t+1} X_t^{(j)}(i)$, and let $\mathfrak{Y}_1 \subseteq \{1, 2, \dots, N\}$ with $|\mathfrak{Y}_1| = Y_1$ be a random set such that $j \in \mathfrak{Y}_1$ iff $\prod_{i=1}^{t+1} X_t^{(j)}(i) = 1$. Conditioned on 1), $Y_1 \geq M$ for sure. For position $i = t+2$, we claim that

$$\begin{aligned} &\mathbf{P}[\sum_{j \in \mathfrak{Y}_1} X_t^{(j)}(i) \geq Y_1 \cdot p_{t,i}(1 - \frac{1}{\rho n}) \,\Big|\, Y_1 \geq M,\ \mathfrak{Y}_1,\ \bigcap_{l=0}^{t-1} \mathcal{E}_l] \\ &\geq 1 - e^{-\frac{1}{2\rho^2 n^2} Y_1} = 1 - e^{-\Omega(\frac{1}{\rho^2 n^2} M)}. \end{aligned} \tag{30}$$

This can be verified with the Chernoff bound. Note that $\sum_{j\in\mathfrak{Y}_1} X_t^{(j)}(i) \geq Y_1 \cdot p_{t,i}(1-\frac{1}{\rho n})$ implies $w_{t,i} \geq p_{t,i}(1-\frac{1}{\rho n})$ conditioned on $Y_1 \geq M$ and $\mathfrak{Y}_1$. Therefore,

$$\mathbf{P}\Big[p_{t+1,i} \geq p_{0,i}(1-\frac{1}{n})^{t+1} \mid Y_1 \geq M,\ \bigcap_{l=0}^{t-1}\mathcal{E}_l\Big] \\ \geq 1 - e^{-\Omega(\frac{1}{\rho^2 n^2}M)}$$

for $i = t+2$.

Now, we discuss the case $i = t+3$ conditioned on $Y_1 \geq M$ and $\sum_{j\in\mathfrak{Y}_1} X_t^{(j)}(t+2) \geq Y_1 \cdot p_{t,i}(1-\frac{1}{\rho n})$. We define $Y_2 := \max\{\sum_{j=1}^N \prod_{r=1}^{t+2} X_t^{(j)}(r),\ M\}$, and the random set $\mathfrak{Y}_2$ containing exactly all those $j \in \mathfrak{Y}_1$ with $\prod_{v=1}^{t+2} X_t^{(j)}(r) = 1$ in the case that $\sum_{j=1}^N \prod_{v=1}^{t+2} X_t^{(j)}(r) \geq M$. In the case that $\sum_{j=1}^N \prod_{r=1}^{t+2} X_t^{(j)}(r) < M$, we add to $\mathfrak{Y}_2$ also the first $M - |\mathfrak{Y}_1|$ many j's from $\mathfrak{Y}_1$ with $\prod_{r=1}^{t+2} X_t^{(j)}(r) = 0$. Then $|\mathfrak{Y}_2| = Y_2$ and M many elite solutions must lie in the set $\{X_t^{(j)} \mid j \in \mathfrak{Y}_2\}$. Moreover, $\{X_t^{(j)} \mid j \in \mathfrak{Y}_2\}$ is exactly the elite sample if $\sum_{j=1}^N \prod_{r=1}^{t+2} X_t^{(j)}(r) \leq M$.

Conditioned on $Y_1 \geq M,\ \mathfrak{Y}_1,\ Y_2 \geq M,\ \mathfrak{Y}_2$, and $\bigcap_{l=0}^{t-1}\mathcal{E}_l$, one obtains similarly that

$$\mathbf{P}\Big[\sum_{j\in\mathfrak{Y}_2} X_t^{(j)}(i) \geq Y_2 \cdot p_{t,i}(1-\frac{1}{\rho n})\Big| \\ Y_1 \geq M,\ \mathfrak{Y}_1,\ Y_2 \geq M,\ \mathfrak{Y}_2,\ \bigcap_{l=0}^{t-1}\mathcal{E}_l\Big] \\ \geq 1 - e^{-\frac{1}{2\rho^2 n^2}Y_2} = 1 - e^{-\Omega(\frac{1}{\rho^2 n^2}M)} \tag{31}$$

for position $i = t+3$. Then, again

$$\mathbf{P}\Big[p_{t+1,i} \geq p_{0,i}(1-\frac{1}{n})^{t+1} \mid Y_1 \geq M,\ \bigcap_{l=0}^{t-1}\mathcal{E}_l\Big] \\ \geq 1 - e^{-\Omega(\frac{1}{\rho^2 n^2}M)}$$

for $i = t+3$.

For the cases $i = t+4, \ldots, n$, we can iteratively define $Y_3, Y_4, \ldots$, and $\mathfrak{Y}_3, \mathfrak{Y}_4, \ldots$ in a similar way. Then, an almost identical argument shows that

$$\mathbf{P}\Big[p_{t+1,i} \geq p_{0,i}(1-\frac{1}{n})^{t+1} \mid Y_1 \geq M,\ \bigcap_{l=0}^{t-1}\mathcal{E}_l\Big] \\ \geq 1 - e^{-\Omega(\frac{1}{\rho^2 n^2}M)}$$

for each $i = t+4, \ldots, n$. As a result, we can conclude that, conditioned on $Y_1 \geq M,\ \bigcap_{l=0}^{t-1}\mathcal{E}_l$, 3) holds for iteration t with a probability at least $(1 - e^{-\Omega(\frac{1}{\rho^2 n^2}M)})^n = 1 - e^{-\Omega(\frac{1}{\rho^2 n^2}M)}$. This concludes the proof of (28).

b) The proof is very similar to the above except for some minor changes in the definitions of the particular random events. Here, we define these $\mathcal{E}_t$'s for iterations $t = 0, 1, \ldots, n-1$ as follows:

1) $\sum_{j=1}^N \prod_{i=1}^{t+1} X_t^{(j)}(i) \geq M$;
2) $p_{t+1,i} \geq 1 - (1 - p_{i-1,i})(1-\rho)^{t+2-i}$ for each $i = 1, \ldots, t+1$;

Again the random event $\bigcap_{l=0}^{n-1}\mathcal{E}_l$ implies that Algorithm 1 reaches the optimal solution in $O(n)$ iterations.

We show that for any $t = 0, 1, \ldots, n-1$,

$$\mathbf{P}[\mathcal{E}_t \mid \bigcap_{l=0}^{t-1}\mathcal{E}_l] \geq 1 - e^{-\Omega(n^\epsilon)}. \tag{32}$$

Note that, conditioned on $\bigcap_{l=0}^{t-1}\mathcal{E}_l$, we have $\prod_{i=0}^{t+1} p_{0,i} \geq \frac{p_{\min}}{n}\prod_{u=1}^{\infty}(1-(1-\rho)^u) = \Omega(\frac{1}{n})$. So, all together,

$$\mathbf{P}(\sum_{j=1}^{N}\prod_{i=1}^{t+1} X_t^{(j)}(i) \geq M \mid \bigcap_{l=0}^{t-1}\mathcal{E}_l) \geq 1 - e^{-(1-\frac{M}{\Omega(\frac{1}{n}N)})^2\frac{N}{2n}} \\ = 1 - e^{-\Omega(n^\epsilon)}.$$

■

References

[1] R. Y. Rubinstein and D. P. Kroese, *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation and machine learning*. Springer Science & Business Media, 2004.

[2] R. Y. Rubinstein, "Optimization of computer simulation models with rare events," *European Journal of Operational Research*, vol. 99, no. 1, pp. 89–112, 1997.

[3] ——, "The cross-entropy method for combinatorial and continuous optimization," *Methodology and computing in applied probability*, vol. 1, no. 2, pp. 127–190, 1999.

[4] M. Dorigo and T. Stützle, *Ant colony optimization*. Cambridge, Massachusetts: A Bradford Book, MIT Press, 2004.

[5] M. Hauschild and M. Pelikan, "An introduction and survey of estimation of distribution algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 3, pp. 111–128, 2011.

[6] M. Zlochin, M. Birattari, N. Meuleau, and M. Dorigo, "Model-based search for combinatorial optimization: A critical survey," *Annals of Operations Research*, vol. 131, no. 1-4, pp. 373–395, 2004.

[7] D. Whitley, "A genetic algorithm tutorial," *Statistics and computing*, vol. 4, no. 2, pp. 65–85, 1994.

[8] H. R. Lourenço, O. C. Martin, and T. Stützle, *Iterated local search*. Springer, 2003.

[9] Z. Wu, "Model-based heuristics for combinatorial optimization: a mathematical study of their asymptotic behavior," Ph.D. dissertation, Institut für Angewandte Stochastik und Operations Research (IASOR), Technical University of Clausthal, 2015.

[10] M. Thomas, *Machine learning*. New Delhi: McGraw Hill Education India, 1997.

[11] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

[12] A. Costa, O. D. Jones, and D. Kroese, "Convergence properties of the cross-entropy method for discrete optimization," *Operations Research Letters*, vol. 35, no. 5, pp. 573–580, 2007.

[13] Z. Wu and M. Kolonko, "Asymptotic properties of a generalized cross entropy optimization algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 5, pp. 658 – 673, 2014.

[14] H. Asoh and H. Mühlenbein, "On the mean convergence time of evolutionary algorithms without selection and mutation," in *Parallel Problem Solving from NaturePPSN III*. Springer, 1994, pp. 88–97.

[15] Z. Wu and M. Kolonko, "Absorption in model-based search algorithms for combinatorial optimization," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*. IEEE, 2014, pp. 1744–1751.

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

[17] J. He and X. Yao, "Drift analysis and average time complexity of evolutionary algorithms," *Artificial Intelligence*, vol. 127, no. 1, pp. 57–85, 2001.

[18] F. Neumann and C. Witt, "Runtime analysis of a simple ant colony optimization algorithm," Departmant of Computer Science, University of Dortmund, Germany, Tech. Rep., 2006.

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

[20] F. Neumann and C. Witt, "Runtime analysis of a simple ant colony optimization algorithm," *Algorithmica*, vol. 54, no. 2, pp. 243–255, 2009.

[21] B. Doerr, F. Neumann, D. Sudholt, and C. Witt, "Runtime analysis of the 1-ant ant colony optimizer," *Theoretical Computer Science*, vol. 412, no. 17, p. 16291644, 2011.

[22] W. J. Gutjahr and G. Sebastiani, "Runtime analysis of ant colony optimization with best-so-far reinforcement," *Methodology & Computing in Applied Probability*, vol. 10, no. 3, pp. 409–433, 2008.

[23] Y. Zhou and J. He, "A runtime analysis of evolutionary algorithms for constrained optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 5, pp. 608–619, 2007.

[24] Y. Zhou, "Runtime analysis of an ant colony optimization algorithm for tsp instances," *Evolutionary Computation IEEE Transactions on*, vol. 13, no. 5, pp. 1083–1092, 2009.

[25] P. S. Oliverto and C. Witt, "Improved time complexity analysis of the simple genetic algorithm," *Theoretical Computer Science*, vol. 605, no. 15, pp. 21–41, 2015.

[26] D. Sudholt and C. Thyssen, "Runtime analysis of ant colony optimization for shortest path problems," *Journal of Discrete Algorithms*, vol. 10, no. 10, pp. 165–180, 2012.

[27] A. Lissovoi and C. Witt, "Runtime analysis of ant colony optimization on dynamic shortest path problems," *Theoretical Computer Science*, vol. 561, pp. 73–85, 2015.

[28] Y. Chen and X. Zou, "Runtime analysis of a multi-objective evolutionary algorithm for obtaining finite approximations of pareto fronts," *Information Sciences*, vol. 262, pp. 62–77, 2014.

[29] F. Neumann, D. Sudholt, and C. Witt, "Analysis of different MMAS ACO algorithms on unimodal functions and plateaus," *Swarm Intelligence*, vol. 3, pp. 35–68, 2009.

[30] T. Chen, K. Tang, G. Chen, and X. Yao, "Analysis of computational time of simple estimation of distribution algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 1, pp. 1–22, 2010.

[31] F. Neumann and C. Witt, "Ant colony optimization and the minimum spanning tree problem," in *Learning and Intelligent Optimization*, V. Maniezzo, R. Battiti, and J.-P. Watson, Eds. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 153–166.

[32] A. M. Sutton, F. Neumann, and S. Nallaperuma, "Parameterized runtime analyses of evolutionary algorithms for the planar euclidean traveling salesperson problem," *Evolutionary Computation*, vol. 22, no. 4, pp. 595–628, 2014.

[33] W. Gasarch, "Classifying problems into complexity classes," *Advances in Computers*, vol. 95, pp. 239–292, 2015.

[34] T. Stützle and H. H. Hoos, "MAX-MIN ant system," *Journal of Future Generation Computer Systems*, vol. 16, pp. 889–914, 2000.

[35] B. Doerr and D. Johannsen, "Refined runtime analysis of a basic ant colony optimization algorithm," in *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, 2007, pp. 480–485.

[36] B. Doerr, "A tight runtime analysis of the $(1+(\lambda, \lambda))$ genetic algorithm on onemax," in *Proceeding of Genetic and Evolunary Computation Conference (GECCO)*. ACM, 2015, pp. 1423–1430.

[37] P. T. D. Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of Operations Research*, vol. 134, no. 1, pp. 19–67, 2005.

[38] H. Mühlenbein, "How genetic algorithms really work: Mutation and hillclimbing." in *Parallel Problem Solving from Nature 2, PPSN-II*, R. Männer and R. Manderick, Eds. North-Holland, Amsterdam, 1992, pp. 15–26.

[39] G. Rudolph, "Convergence properties of evolutionary algorithms," Ph.D. dissertation, Verlag Dr. Kovac, 1997.

[40] C. Witt, "Revised analysis of the (1+1) EA for the minimum spanning tree problem," in *Proceedings of the 2014 Conference on Genetic and Evolutionary Computation*. ACM, 2014, pp. 509–516.

[41] M. Mitzenmacher and E. Upfal, *Probability and computing: randomized algorithms and probabilistic analysis*. Cambridge University Press, 2005.

[42] P. J. Boland and F. Proschan, "The reliability of k out of n systems," *Annals of Probability*, vol. 11, no. 3, pp. 760–764, 1983.

[43] L. J. Gleser, "On the distribution of the number of successes in independent trials," *Annals of Probability*, vol. 3, no. 1, pp. 182–188, 1975.