Theoretical Study of Optimizing Rugged Landscapes with the cGA

Tobias Friedrich[1], Timo Kötzing[1], Frank Neumann[2], and Aishwarya Radhakrishnan[1]

[1] Hasso Plattner Institute, University of Potsdam, Potsdam, Germany
friedrich@hpi.de, timo.koetzing@hpi.de, aishwarya.radhakrishnan@hpi.de
[2] School of Computer Science, The University of Adelaide, Adelaide, Australia
frank.neumann@adelaide.edu.au

Abstract. Estimation of distribution algorithms (EDAs) provide a distribution-based approach for optimization which adapts its probability distribution during the run of the algorithm. We contribute to the theoretical understanding of EDAs and point out that their distribution approach makes them more suitable to deal with rugged fitness landscapes than classical local search algorithms.
Concretely, we make the OneMax function rugged by adding noise to each fitness value. The cGA can nevertheless find solutions with $n(1-\varepsilon)$ many 1s, even for high variance of noise. In contrast to this, RLS and the (1+1) EA, with high probability, only find solutions with $n(1/2 + o(1))$ many 1s, even for noise with small variance.

Keywords: Estimation-of-distribution algorithms · Compact genetic algorithm · Random local search · Evolutionary algorithms · Run time analysis · Theory

1   Introduction

Local search [1], evolutionary algorithms [9] and other types of search heuristics have found applications in solving classical combinatorial optimization problems as well as challenging real-world optimization problems arising in areas such as mine planning and scheduling [18,20] and renewable energy [24,19].

Local search techniques perform well if the algorithm can achieve improvement through local steps whereas other more complex approaches such as evolutionary algorithms evolving a set of search points deal with potential local optima by diversifying their search and allowing to change the current solutions through operators such as mutation and crossover. Other types of search heuristics, such as estimation of distribution algorithms [21] and ant colony optimization [8], sample their solutions in each iteration from a probability distribution that is adapted based on the experience made during the run of the algorithm. One of the key questions that arises is when to favour distribution-based algorithms over search point-based methods. We will investigate this in the context of rugged landscapes that are obtained by stochastic perturbation.

Real-world optimization problems are often rugged with many local optima and quantifying and handling rugged landscapes is an important topic when using search heuristics [17,22,3]. Small details about the chosen search point can lead to a rugged fitness landscapes even if the underlying problem has a clear fitness structure which, by itself, would allow local search techniques to find high quality solution very quickly.

In this paper we model a rugged landscape with underlying fitness structure via OneMax,[3] where each search point is perturbed by adding an independent sample from some given distribution D. We denote the resulting (random) fitness function OM_D.

Note that this setting of D-rugged OneMax is different from noisy One-Max (with so-called additive posterior noise) [13,11,23,2,12] in that evaluating a search point multiple times does not lead to different fitness values (which then could be averaged, implicitly or explicitly) to get a clearer view of the underlying fitness signal. Note that another setting without reevaluation (but on a combinatorial path problem and for an ant colony algorithm, with underlying non-noisy ground truth) was analyzed in [14].

In this paper we consider as distribution the normal distribution as well as the geometric distribution. Since all search points get a sample from the same distribution added, the mean value of the distribution is of no importance (no algorithm we consider makes use of the absolute fitness value, only of relative values). Mostly important is the steepness of the tail, and in this respect the two distributions are very similar.

An important related work, [10] discusses what impact the *shape* of the chosen noise model has on optimization of noisy OneMax. They find that steep tails behave very differently from uniform tails, so for this first study we focus on two distributions with steep tails, which we find to behave similarly.

As a first algorithm, which was found to perform very well under noise [2], we consider the *compact genetic algorithm* (cGA), an estimation of distribution algorithm which has been subject to a wide range of studies in the theoretical analysis of estimation of distribution algorithms [12,6,5,16]. See Section 2 for an exposition of the algorithm.

In Theorem 4 we show the cGA to be efficient on $\mathcal{N}(0, \sigma^2)$-rugged OneMax, even for arbitrarily large values of σ^2 (at the cost of a larger run time bound). Note that, since the optimum is no longer guaranteed to be at the all-1s string (and the global optimum will be just a rather random search point with a lot of 1s, we only consider the time until the cGA reaches $n(1-\varepsilon)$ many 1s (similar to [23]). The idea of the proof is to show that, with sufficiently high probability, no search point is evaluated twice in a run of the cGA; then the setting is identical to OneMax with additive posterior noise and we can cite the corresponding theorem from the literature [12]. Thus, working with a distribution over the search space and adapting it during the search process leads to a less coarse-

---

[3] OneMax is the well-studied pseudo-Boolean test function mapping $x \in \{0,1\}^n$ to $\sum_{i=1}^{n} x_i$.

grained optimization process and presents a way to deal effectively with rugged fitness landscapes.

We contrast this positive result with negative results on search point based methods, namely random local search (RLS, maintaining a single individual and flipping a single bit each iteration, discarding the change if it worsened the fitness) and the so-called (1+1) EA (which operates like RLS, but instead of flipping a single bit, each bit is flipped with probability $1/n$), as well as with Random Search (RS, choosing a uniformly random bit string each iteration).

We first consider random local search on $\mathcal{N}(0, \sigma^2)$-rugged OneMax. Theorem 5 shows that, for noise in $\Omega(\sqrt{\log n})$, RLS will not make it further than half way from the random start to the OneMax-optimum and instead get stuck in a local optimum. The proof computes, in a rather lengthy technical lemma, the probability that a given search point has higher fitness than all of its neighbors.

Going a bit more into detail about what happens during the search, we consider the geometric distribution. In Theorem 6 we prove that, even for constant variance, with high probability the algorithm is stuck after transitioning to a new search point at most $\log^2(n)$ many times. This means that essentially no progress is made over the initial random search point, even for small noise values! The proof proceeds by showing that successive accepted search points have higher and higher fitness values; in fact, the fitness values grow faster than the number of 1s in the underlying bit string. Since every new search point has to have higher fitness than the previous, it quickly is unfeasible to find even better search points (without going significantly higher in the 1s of the bit string).

In Theorem 7 we translate the result for RLS to the (1+1) EA. We require small but non-constant variance of D to make up for the possibility of the (1+1) EA to jump further, but otherwise get the result with an analogous proof. To round off these findings, we show in Theorem 8 that Random Search has a bound of $O(\sqrt{n \log n})$ for the number of 1s found within a polynomial number of iterations.

In Section 7 we give an experimental impression of the negative results. We depict that, within n^2 iterations, the proportion of 1s in the best string found is decreasing in n for all algorithms RLS, (1+1) EA and RS, where RS is significantly better than the (1+1) EA, RLS being the worst.

This paper proceeds with some preliminaries on the technical details regarding the algorithms and problems considered. After the performance analyses of the different algorithms in Section 3 through 6 and the experimental evaluation in Section 7, we conclude in Section 8. We defer some technical lemmas to Appendix **??**.

2   Algorithms and Problem Setting

In this section we define the D-rugged OneMax problem and describe all the algorithms which we are analyzing in this paper. Random local search (RLS) on a fitness function f is given in Algorithm 1. RLS samples a point uniformly at random and at each step creates an offspring by randomly flipping a bit. At the end of each iteration it retains the best bit string available.

Algorithm 1: RLS on fitness function f

---

1 Choose $x \in \{0,1\}^n$ uniformly at random;
2 **while** *stopping criterion not met* **do**
3 $y \leftarrow$ flip one bit of x chosen uniformly at random;
4   **if** $f(y) \geq f(x)$ **then** $x \leftarrow y$;

The (1+1) EA on a fitness function f is given in Algorithm 2. The difference between RLS and (1+1) EA is that (1+1) EA creates an offspring by flipping each bit with probability $1/n$.

Algorithm 2: (1+1) EA on fitness function f

---

1 Choose $x \in \{0,1\}^n$ uniformly at random;
2 **while** *stopping criterion not met* **do**
3 $y \leftarrow$ flip each bit of x independently with probability $1/n$;
4   **if** $f(y) \geq f(x)$ **then** $x \leftarrow y$;

The cGA on a fitness function f is given in Algorithm 3. This algorithm starts with two bit strings which have the probability of $1/2$ for each of their bit to be 1. After each step this probability is updated based on the best bit string encountered.

Algorithm 3: The compact GA on fitness function f

---

1 $t \leftarrow 0$, K $\leftarrow$ initialize;
2 $p_{1,t} \leftarrow p_{2,t} \leftarrow \cdots \leftarrow p_{n,t} \leftarrow 1/2$;
3 **while** *termination criterion not met* **do**
4   **for** $i \in \{1, \ldots, n\}$ **do**
5 $x_i \leftarrow 1$ with probability $p_{i,t}$, $x_i \leftarrow 0$ else;
6   **for** $i \in \{1, \ldots, n\}$ **do**
7 $y_i \leftarrow 1$ with probability $p_{i,t}$, $y_i \leftarrow 0$ else;
8   **if** $f(x) < f(y)$ **then** swap x and y;
9   **for** $i \in \{1, \ldots, n\}$ **do**
10   **if** $x_i > y_i$ **then** $p_{i,t+1} \leftarrow p_{i,t} + 1/K$;
11   **if** $x_i < y_i$ **then** $p_{i,t+1} \leftarrow p_{i,t} - 1/K$;
12   **if** $x_i = y_i$ **then** $p_{i,t+1} \leftarrow p_{i,t}$;
13 $t \leftarrow t + 1$;

2.1 D-Rugged OneMax

To give a simple model for a rugged landscape with underlying gradient, we use a randomly perturbed version of the well-studied OneMax test function. We fix a dimension $n \in \mathbb{N}$ and a random distribution D. Then we choose, for every $x \in \{0,1\}^n$, a random distortion y_x from the distribution D. We define a D-rugged OneMax function as $\mathrm{OM}_D \colon \{0,1\}^n \to \mathbb{R} := x \mapsto \|x\|_1 + y_x$ where $\|x\|_1 := |\{i \mid x_i = 1\}|$ is the number of 1s in x.

In the following sections we show that the cGA optimizes even very rugged distortions of OM_D efficiently, while RLS will get stuck in a local optimum.

3 Performance of the cGA

Let $D \sim N(0, \sigma^2)$. The following is Lemma 5 from [12], which shows that, while optimizing OM_D, the probability that marginal probabilities falls a constant less than $1/2$ is superpolynomially small in n. Note that here and in all other places in this paper, we give probabilities that range over the random choices of the algorithm as well as the random landscape of the instance.

Lemma 1 *Let $\varepsilon \in (0,1)$ and define*

$$M_\varepsilon = \left\{ p \in [0.25, 1]^n \; \middle| \; \sum_{i=1}^n p_i \leq n(1-\varepsilon) \right\}.$$

Then

$$\max_{p,q \in M_\varepsilon} \sum_{i=1}^n (2p_i q_i - p_i - q_i) \leq -n\varepsilon/2.$$

Lemma 2 ([12],[7]) *Let $D \sim N(0,\sigma^2)$. Consider the cGA optimizing OM_D with $\sigma^2 > 0$. Let $0 < a < 1/2$ be an arbitrary constant and $T' = \min\{t \geq 0 : \exists i \in [n], p_{i,t} \leq a\}$. If $K = \omega(\sigma^2 \sqrt{n} \log n)$, then for every polynomial $\mathrm{poly}(n)$, n sufficiently large, $\Pr(T' < \mathrm{poly}(n))$ is superpolynomially small.*

Theorem 3 *Let $\varepsilon \in (0,1)$ and define*

$$M_\varepsilon = \left\{ p \in [0.25, 1]^n \; \middle| \; \sum_{i=1}^n p_i \leq n(1-\varepsilon) \right\}.$$

Let $S \subseteq \{0,1\}^n$ be a random multi set of polynomial size (in n), where each member is drawn independently according to some marginal probabilities from M_ε. Then the probability that the multi set contains two identical bit strings is $2^{-\Omega(n\varepsilon)}$.

Proof. Let $x, y \in S$ be given, based on marginal probabilities $p, q \in M_\varepsilon$, respectively. Using Lemma 1 in the last step, we compute

$$P(x = y) = \prod_{i=1}^n P(x_i = y_i)$$

$$= \prod_{i=1}^{n} (p_i q_i + (1 - p_i)(1 - q_i))$$

$$= \prod_{i=1}^{n} (1 - p_i - q_i + 2p_i q_i)$$

$$\leq \prod_{i=1}^{n} \exp(-p_i - q_i + 2p_i q_i)$$

$$\leq \exp\left(\sum_{i=1}^{n} -p_i - q_i + 2p_i q_i\right)$$

$$\underset{\text{Lemma 1}}{\leq} \exp(-n\varepsilon/2).$$

Since there are only polynomially many pairs of elements from S, we get the claim by an application of the union bound.

$\square$

From the preceding theorem we can now assume that the cGA always samples previously unseen search points. We can use [12, Lemma 4] which gives an additive drift of $O(\sqrt{n}/(K\sigma^2))$ in our setting.

Now we can put all ingredients together and show the main theorem of this section.

Theorem 4 *Let $D \sim N(0, \sigma^2)$, $\sigma^2 > 0$ and let ε be some constant. Then the cGA with $K = \omega(\sigma^2 \sqrt{n} \log n)$ optimizes OM_D up to a fitness of $n(1 - \varepsilon)$ within an expected number of $O(K\sqrt{n}\sigma^2)$ iterations.*

Proof. We let $X_t = \sum_{i=1}^{n} p_{i,t}$ be the sum of all the marginal probabilities at time step t of the cGA. Using Lemma 2, we can assume that (for polynomially many time steps) the cGA has marginal probabilities of at least 0.25. Now we can employ Lemma 3 to see that the cGA does not sample the same search point twice in a polynomial number of steps. Thus, as mentioned, we can use [12, Lemma 4] to get an additive drift of $O(\sqrt{n}/(K\sigma^2))$ as long as X_t does not reach a value of $n(1 - \varepsilon)$.

The maximal value of X_t is n, so we can use an additive drift theorem that allows for overshooting [15, Theorem 3.7] to show the claim.

$\square$

4   Performance of RLS

In this section we show that RLS cannot optimize rugged landscapes even for small values of ruggedness. We show that RLS will not find a solution with more than $3n/4$ ones with high probability. This implies that RLS will get stuck in a local optimum with a high probability because there are exponentially many points with number of ones more than $3n/4$ and the probability that none of this points have noise more than the noise associated with the best solution found by RLS is very low.

Theorem 5 *Let $D \sim N(0, \sigma^2)$. Let $\sigma^2 \geq 4\sqrt{2\ln(n+1)}$. Then there is a constant $c < 1$ such that RLS optimizing OM_D will reach a solution of more than $3n/4$ many 1s (within any finite time) will have a probability of at most c.*

Proof. We consider the event A_0 that RLS successfully reached a fitness of $3n/4$ starting from an individual with at most $n/2$ many 1s.

With probability at least $1/2$ the initial search point of RLS has at most $n/2$ many 1s.

We define, for each level $i \in \{n/2, n/2 + 3, n/2 + 6, \ldots, 3n/4 - 3\}$, the first *accepted* individual x_i which RLS found on that level. For the event A_0 to hold, it must be the case that all x_i are *not* local optima. Any search points in the neighborhood of x_i sampled previous to the encounter of x_i will have a value less than x_i (since x_i is accepted) and the decision of whether x_i is a local optimum depends only on the $k < n$ so far not sampled neighbors. Since two different x_i have a Hamming distance of at least 3, these neighbors are disjoint sets (for different i) and their noises are independent.

For any point x to be a local optimum, it needs to have a higher fitness than any of its at most n neighbors. We assume pessimistically that all neighbors have one more 1 in the bit string and compute a lower bound on the probability that the random fitness of x is higher than the random fitness of any neighbor by bounding the probability that a Gaussian random variable is larger than the largest of n Gaussian random variables plus 1. By scaling, this is the probability that some $\mathcal{N}(0, 1)$-distributed random variable is higher than the maximum of n independent $\mathcal{N}(1/\sigma^2, 1)$-distributed random variables.

Using the symmetry of the normal distribution, this is equivalent to the probability that some $\mathcal{N}(1/\sigma^2, 1)$-distributed random variable is *less* than the *minimum* of n $\mathcal{N}(0, 1)$-distributed random variables. This is exactly the setting of Lemma **??**, where we pick $c = 1/\sigma^2$. Plugging in our bound for σ^2, we get a probability of $\Omega(1/n)$ that an arbitrary point in our landscape is a local optimum.

Thus we get with constant probability that one of the $\Theta(n)$ many x_i is a local optimum. With this constant probability, event A_0 cannot occur, as desired.

$\square$

4.1 Performance of RLS – a detailed look

We now want to give a tighter analysis of RLS on rugged ONEMAX by showing that, in expectation, the noise of new accepted search points is growing. For the analysis, we will switch to a different noise model: We now assume our noise to be $\mathrm{Geo}(p)$-distributed, for some $p \leq 1/2$. We believe that a similar analysis is also possible for normal-distributed noise, but in particular with much more complicated Chernoff bounds.

Theorem 6 *Let $p \leq 1/2$ and let $D \sim \mathrm{Geo}(p)$. Then, for all c, the probability that RLS optimizing D-rugged ONEMAX will transition to a better search point more than $log^2(n)$ times is $O(n^{-c})$.*

In particular, in this case, RLS does not make progress of $\Omega(n)$ over the initial solution and does not find the optimum.

Proof. We consider the run of RLS optimizing D-rugged ONEMAX and denote with X_t the *noise* of the t-th *accepted* search point. We know that $X_0 \sim D$ and each next point has to be larger than at least the previous search point minus 1: in each iteration either a 0 bit or a 1 bit is flipped and RLS accepts the new search point only if its fitness value is greater than or equal to the previous search point.

We will show that X_t is, in expectation, growing. Furthermore we will show that, with high probability, for $t = \log^2(n)$ we have that $X_t \geq \log^2(n)/2$. We finish the proof by showing that, with a search point with such a noise value, it is very unlikely to find a better search point.

Note that, for all t, we have that the distribution of X_{t+1} is the distribution of D conditional on being at least $X_t - 1$ if a 0 was flipped to a 1 and $X_t + 1$ otherwise. Pessimistically assuming the first case and since D is memory-less, we get $X_{t+1} \sim X_t - 1 + D$. In particular, since $E[D] = \frac{1}{p}$, we have $E[X_{t+1}] \geq E[X_t] + 1/p - 1$. Inductively we get

$$E[X_t] \geq E[X_0] + \frac{t}{p} - t. \tag{1}$$

Let the geometric random variable attached with each X_t be D_t and we have $X_t \sim X_{t-1} - 1 + D_t$, therefore $X_t \sim \sum_{i=0}^{t} D_i - t$. This implies $P(X_t \leq (\frac{t+1}{p} - t)/2)$ is nothing but $P(\sum_{i=0}^{t} D_i \leq (\frac{t+1}{p} + t)/2)$. By using Chernoff bounds for the sum of independent geometric random variables [4, Theorem 1.10.32] and by letting $\delta = \frac{1}{2} - \frac{tp}{2(t+1)}$ we have,

$$P\left(X_t \leq \left(\frac{t+1}{p} - t\right)/2\right) = P\left(\sum_{i=0}^{t} D_i \leq \left(\frac{t+1}{p} + t\right)/2\right)$$

$$= P\left(\sum_{i=0}^{t} D_i \leq (1-\delta)\frac{t+1}{p}\right)$$

$$\leq \exp\left(-\frac{\delta^2(t+1)}{2 - \frac{4\delta}{3}}\right).$$

Since $p \leq \frac{1}{2}$, we have $\delta \geq \frac{1}{4}$. Therefore,

$$P\left(X_t \leq \left(\frac{t+1}{p} - t\right)/2\right) \leq \exp\left(-\frac{\delta^2(t+1)}{2 - \frac{4\delta}{3}}\right) \leq \exp\left(-\frac{3t}{80}\right)$$

When $t = \log^2(n)$,

$$P\left(X_t \leq \left(\frac{t+1}{p} - t\right)/2\right) \leq \exp\left(-\frac{3t}{80}\right) = n^{-\frac{3}{80}\log(n)}.$$

Assume that we sampled a search point with noise at least $m = \frac{t+1}{2p} - \frac{t}{2}$, where $t = \log^2(n)$. For a neighbor of the current search point to have higher fitness it

should have at least $m-1$ or $m+1$ noise, depending on whether it has an extra 1 bit or an extra zero bit. The probability for this to happen is,

$$
\begin{aligned}
P(D \geq m+1) &\leq P(D \geq m-1) \\
&= p(1-p)^{-2}(1-p)^m \\
&\leq e^{\frac{1}{2}} n^{-\frac{1}{4}\log(n)}.
\end{aligned}
$$

Using this, we will show that once a search point with noise at least m is sampled, the probability that at least one of the neighbours is of higher fitness is $O\left(n^{1-\frac{\log(n)}{4}}\right)$. Let $D_{m_1}, \ldots, D_{m_n}$ denote the random geometric noise associated with the neighbors of the current search point with at least noise m. Then probability that at least one of the neighbours is of higher fitness is

$$
\begin{aligned}
&\leq P(D_{m_1} \geq m-1 \cup \cdots \cup D_{m_n} \geq m-1) \\
&\leq \sum_{i=1}^{n} P(D_{m_i} \geq m-1) \\
&\leq e^{\frac{1}{2}} n^{1-\frac{\log(n)}{4}}.
\end{aligned}
$$

$\square$

5   Performance of the (1+1) EA

In this section we extend the analysis given for RLS in Theorem 6 to the (1+1) EA.

Theorem 7 *Let $p \leq 1/(2\log(n))$ and let $D \sim \text{Geo}(p)$. Then, for all $c > 0$ and $k \in \mathbb{N}$, the probability that the (1+1) EA optimizing D-rugged ONEMAX will transition to a better search point more than $\log^2(n)$ times within n^k steps is $O(n^{-c})$.*

In particular, the probability that the (1+1) EA makes progress of $\Omega(n)$ over the initial solution within n^k steps is $O(n^{-c})$ and thus does not find the optimum.

Proof. We first show that, for $c > 0$, in any iteration the new accepted search point by (1+1) EA does not have more than $ck\log(n)-1$ ones than the previous accepted point with probability at least $1 - O(n^{-c})$. Then we assume the worst case scenario that the new search point has $ck\log(n)-1$ more ones than the previous search point to proceed with the proof similar to Theorem 6 for RLS.

Let X denote the number of bit flips happened to get the current search point. Then $X \sim \text{Bin}(n, 1/n)$ (which has an expectation of 1). If the current search point has $ck\log(n)-1$ more ones than the previous search point then X has to be at least $ck\log(n)-1$. By a multiplicative Chernoff bound for the sum of independent Bernoulli trails, if $\delta = ck\log(n)-2$,

$$
P(X \geq ck\log(n)-1) = P(X \geq 1+\delta)
$$

$$\leq \exp\left(-\frac{(ck\log(n)-2)^2}{ck\log(n)}\right)$$

$$\leq \frac{e^4}{n^{ck}}.$$

A union bound over n^k iterations gives that the probability that within any of these iterations the (1+1) EA jumps further than $ck\log(n)-1$ is $O(n^{-c})$.

Consider now the run of the (1+1) EA optimizing $D-$rugged ONEMAX and let X_t denote the noise associated with the $t-$th accepted search point. Similar to Theorem 6, we now show the following. (1) X_t grows in expectation. (2) For $t = \log^2(n)$, with high probability $X_t \geq \log^2(n)/2$. (3) If we have a search point with noise greater than $\log^2(n)/2$, the probability to find a better search point within n^k steps is very low. We pessimistically assume that the t-th accepted search point has $\log(n)-1$ ones more than the previous search point. Since D is memory-less, we get $X_t \sim X_{t-1} - \log(n) + 1 + D$. This inductively along with the fact that $X_0 = D$ implies,

$$E[X_t] \geq \frac{t+1}{p} - t\log(n) + t.$$

Let D_t be the geometric random variable associated with X_t. Then we have $X_t \sim X_{t-1} - \log(n) + 1 + D_t$, which is $X_t \sim \sum_{i=0}^{t} D_i - t\log(n) + t$. If we let $\delta = \frac{1}{2} + \frac{tp}{2(t+1)} - \frac{tp\log(n)}{2(t+1)}$, $a_t = \left(\frac{t+1}{p} - t\log(n) + t\right)/2$ and use Chernoff bounds for the sum of independent geometric random variables [4, Theorem 1.10.32] we have,

$$P\left(X_t \leq a_t\right) = P\left(\sum_{i=0}^{t} D_i \leq \frac{t+1}{2p} + \frac{t\log(n)}{2} - \frac{t}{2}\right)$$

$$= P\left(\sum_{i=0}^{t} D_i \leq (1-\delta)\frac{t+1}{p}\right)$$

$$\leq \exp\left(-\frac{\delta^2(t+1)}{2 - \frac{4\delta}{3}}\right).$$

Since $p \leq \frac{1}{2\log(n)}$, we have $\delta \geq \frac{1}{4}$. Therefore,

$$P\left(X_t \leq \left(\frac{t+1}{p} - t\log(n) + t\right)/2\right) \leq \exp\left(-\frac{\delta^2(t+1)}{2 - \frac{4\delta}{3}}\right) \leq \exp\left(-\frac{3t}{80}\right).$$

When $t = \log^2(n)$,

$$P\left(X_t \leq \left(\frac{t+1}{p} - t\log(n) + t\right)/2\right) \leq \exp\left(-\frac{3t}{80}\right)$$

$$= n^{-\frac{3}{80}\log(n)}.$$

Now assume that we sampled a search point with noise at least $m = \frac{t+1}{2p} - \frac{t\log(n)}{2} + \frac{t}{2}$, where $t = \log^2(n)$. As we have seen at any given iteration the probability that the standard mutation operator flips more than $ck\log(n) - 1$ bits is very low, we will again analyze the worst case scenario. For a neighbor of the current search point with at most $ck\log(n) - 1$ to have higher fitness it should have at least $m - ck\log(n)$ noise. The probability for this to happen is,

$$P(D \geq m - ck\log n + 1) = p(1-p)^{m-ck\log(n)} \leq e^{-p(m-ck\log(n))} = e^{\frac{ck}{2}}n^{-\frac{1}{4}\log n}$$

For a given k, let $D_{m_1}, \ldots, D_{m_{n^k}}$ denote the random geometric noise associated with n^k neighbors of the current search point with at least noise m. Then probability that within n^k steps at least one of the neighbours will be of higher fitness is at most

$$P\left(\bigcup_{i=1}^{n}(D_{m_i} \geq m - ck\log(n) + 1)\right) \leq \sum_{i=1}^{n^k} P(D_{m_i} \geq m - ck\log(n) + 1)$$
$$\leq e^{\frac{ck}{2}}n^{k-\frac{\log(n)}{4}}.$$

$\square$

6   Performance of Random Search

For comparison with the performance of RLS and the (1+1) EA, we briefly consider Random Search (RS) in this section. We state the theorem that Random Search has a bound of $O(\sqrt{n\log n})$ for the number of 1s found within a polynomial number of iterations and can be proved by Chernoff bounds.

Theorem 8 *Let $c > 0$ be given and $t \geq 1$. Then the bit string with the most number of 1s found by Random Search within $t \leq n^k$ iterations, choosing a uniformly random bit string each iteration, has at most $n/2 + O(\sqrt{n\log n})$ 1s with probability $1 - O(n^{-c})$.*

7   Experimental Evaluation

In this section we empirically analyze the performance of the cGA, the RLS, the (1+1) EA and the Random Search algorithms on the rugged ONEMAX with two different noise models. We considered noise sampled from the normal distribution with mean zero and variance 5 and another noise model sampled from the geometric distribution with variance 5. From the results we can see that after n^2 iterations, where n is length of the bit string, the RLS and the (1+1) EA does not sample a search point with more than 60% of 1s but the cGA with $K = \sqrt{n}\log(n)$ always finds the optimum. The plot in Figure 1 is mean of 100 independent runs of each algorithm for bit string lengths 100 to 1000 with step size 100. To have closer look at the performance of the other algorithms, performance of the cGA(straight line at 100) is removed from the plot.
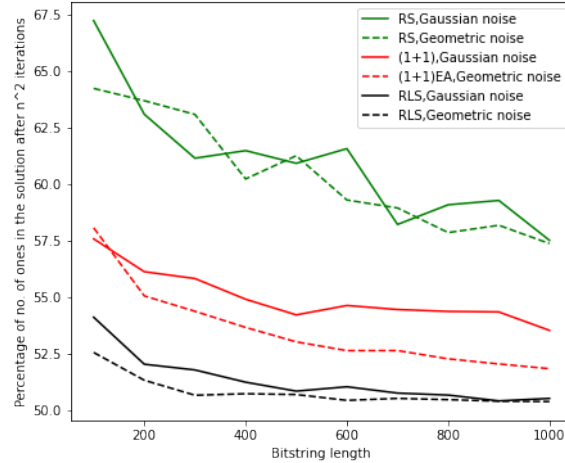
Fig. 1. Percentage of ones in the sampled search point after n^2 iterations of Random Search, (1+1) EA and RLS on optimizing rugged OneMax function with Normal and Geometric noise both having variance of 5. Performance of cGA which is a straight line at 100% is omitted.

Each time the first search point was set to have 50% of 1s. As n increases the percentage of ones in the search point sampled by RLS and (1+1) EA after n^2 iteration tends to 50% and it tends to less than 60% in case of random search.

8   Conclusion

Rugged fitness landscapes appear in many real-world problems and may lead to algorithms getting stuck in local optima. We investigated this problem in this paper for the rugged OneMax problem which is obtained through a noisy OneMax fitness function where each search point is only evaluated once. We have shown that RLS and the (1+1) EA can only achieve small improvements on an initial solution chosen uniformly at random. In contrast to this the cGA is able to improve solution quality significantly until it is (almost) optimal. Our experimental investigations show this behaviour for realistic input sizes and also point out that RLS and the (1+1) EA perform significantly worse than random search on the rugged OneMax problem.

Acknowledgements

References

1. Aarts, E., Lenstra, J.K.: Local Search in Combinatorial Optimization. John Wiley & Sons, Inc. (1997)
2. Aishwaryaprajna, Rowe, J.E.: Noisy combinatorial optimisation by evolutionary algorithms. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion. p. 139–140. GECCO '19 (2019). https://doi.org/10.1145/3319619.3321955, `https://doi.org/10.1145/3319619.3321955`
3. Daolio, F., Liefooghe, A., Verel, S., Aguirre, H., Tanaka, K.: Problem Features versus Algorithm Performance on Rugged Multiobjective Combinatorial Fitness Landscapes. Evolutionary Computation **25**, 555–585 (2017). https://doi.org/10.1162/evco_a_00193, `https://doi.org/10.1162/evco_a_00193`
4. Doerr, B.: Probabilistic tools for the analysis of randomized optimization heuristics. In: Doerr, B., Neumann, F. (eds.) Theory of Evolutionary Computation, p. 1–87. Springer International Publishing (2019). https://doi.org/10.1007/978-3-030-29414-4_1, `http://dx.doi.org/10.1007/978-3-030-29414-4_1`
5. Doerr, B.: The runtime of the compact genetic algorithm on jump functions. Algorithmica **83**, 3059–3107 (2021). https://doi.org/10.1007/s00453-020-00780-w, `https://doi.org/10.1007/s00453-020-00780-w`
6. Doerr, B., Zheng, W.: From understanding genetic drift to a smart-restart parameter-less compact genetic algorithm. In: Coello, C.A.C. (ed.) GECCO '20: Genetic and Evolutionary Computation Conference. pp. 805–813 (2020). https://doi.org/10.1145/3377930.3390163, `https://doi.org/10.1145/3377930.3390163`
7. Doerr, B., Zheng, W.: Sharp bounds for genetic drift in estimation of distribution algorithms. IEEE Transactions on Evolutionary Computation **24**(6), 1140–1149 (2020). https://doi.org/10.1109/TEVC.2020.2987361
8. Dorigo, M., Stützle, T.: Ant colony optimization. MIT Press (2004)
9. Eiben, A.E., Smith, J.E.: Introduction to Evolutionary Computing, Second Edition. Natural Computing Series, Springer (2015)
10. Friedrich, T., Kötzing, T., Krejca, M.S., Sutton, A.M.: Graceful scaling on uniform versus steep-tailed noise. In: Handl, J., Hart, E., Lewis, P.R., López-Ibáñez, M., Ochoa, G., Paechter, B. (eds.) Proc. of PPSN'16. Lecture Notes in Computer Science, vol. 9921, pp. 761–770. Springer (2016). https://doi.org/10.1007/978-3-319-45823-6_71, `https://doi.org/10.1007/978-3-319-45823-6_71`
11. Friedrich, T., Kötzing, T., Krejca, M., Sutton, A.M.: The benefit of sex in noisy evolutionary search (2015)
12. Friedrich, T., Kötzing, T., Krejca, M.S., Sutton, A.M.: The compact genetic algorithm is efficient under extreme gaussian noise. Transactions on Evolutionary Computation **21**, 477–490 (2017). https://doi.org/10.1109/TEVC.2016.2613739
13. Gießen, C., Kötzing, T.: Robustness of populations in stochastic environments. Algorithmica **75**, 462–489 (2016). https://doi.org/10.1007/s00453-015-0072-0, `https://doi.org/10.1007/s00453-015-0072-0`
14. Horoba, C., Sudholt, D.: Ant colony optimization for stochastic shortest path problems. In: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation. p. 1465–1472. GECCO '10 (2010). https://doi.org/10.1145/1830483.1830750, `https://doi.org/10.1145/1830483.1830750`

15. Krejca, M.S.: Theoretical analyses of univariate estimation-of-distribution algorithms. doctoralthesis, Universität Potsdam (2019). https://doi.org/10.25932/publishup-43487
16. Lengler, J., Sudholt, D., Witt, C.: The complex parameter landscape of the compact genetic algorithm. Algorithmica **83**, 1096–1137 (2021). https://doi.org/10.1007/s00453-020-00778-4, `https://doi.org/10.1007/s00453-020-00778-4`
17. Malan, K.M., Engelbrecht, A.P.: Quantifying ruggedness of continuous landscapes using entropy. In: IEEE Congress on Evolutionary Computation. pp. 1440–1447 (2009)
18. Myburgh, C., Deb, K.: Evolutionary algorithms in large-scale open pit mine scheduling. In: GECCO. pp. 1155–1162. ACM (2010)
19. Neshat, M., Alexander, B., Wagner, M.: A hybrid cooperative co-evolution algorithm framework for optimising power take off and placements of wave energy converters. Inf. Sci. **534**, 218–244 (2020)
20. Osada, Y., While, R.L., Barone, L., Michalewicz, Z.: Multi-mine planning using a multi-objective evolutionary algorithm. In: IEEE Congress on Evolutionary Computation. pp. 2902–2909 (2013)
21. Pelikan, M., Hauschild, M., Lobo, F.G.: Estimation of distribution algorithms. In: Kacprzyk, J., Pedrycz, W. (eds.) Handbook of Computational Intelligence, pp. 899–928 (2015)
22. Poursoltan, S., Neumann, F.: Ruggedness quantifying for constrained continuous fitness landscapes. In: Evolutionary Constrained Optimization, pp. 29–50. Springer (2015)
23. Prugel-Bennett, A., Rowe, J., Shapiro, J.: Run-time analysis of population-based evolutionary algorithm in noisy environments. In: Proceedings of the 2015 8th Conference on Foundations of Genetic Algorithms. p. 69–75. FOGA '15 (2015). https://doi.org/10.1145/2725494.2725498, `https://doi.org/10.1145/2725494.2725498`
24. Tran, R., Wu, J., Denison, C., Ackling, T., Wagner, M., Neumann, F.: Fast and effective multi-objective optimisation of wind turbine placement. In: GECCO. pp. 1381–1388. ACM (2013)