Fixed-target runtime analysis of the (1 + 1) EA with resampling (student workshop paper)

Dmitry Vinokurov, Maxim Buzdalov, Arina Buzdalov, Benjamin Doerr, Carola Doerr

Fixed-Target Runtime Analysis of the $(1 + 1)$ EA with Resampling

Dmitry Vinokurov
ITMO University
Saint Petersburg, Russia

Maxim Buzdalov
ITMO University
Saint Petersburg, Russia

Arina Buzdalova
ITMO University
Saint Petersburg, Russia

Benjamin Doerr
École Polytechnique, CNRS, LIX
Palaiseau, France

Carola Doerr
Sorbonne Université, CNRS, LIP6
Paris, France

ABSTRACT

We conduct a fixed-target runtime analysis of (1 + 1) EA with resampling on the OneMax and BinVal problems. For OneMax, our fixed-target upper bound refines the previously known bound. Our fixed-target lower bound for OneMax is the first of this kind. We also consider linear functions and show that the traditional approaches via drift analysis cannot easily be extended to yield fixed-target results. However, for the particular case of BinVal, a relatively precise fixed-target bound is obtained.

CCS CONCEPTS

• **Theory of computation → Theory of randomized search heuristics**.

KEYWORDS

runtime analysis, fixed-target analysis, drift analysis, resampling

1 INTRODUCTION

Theory of evolutionary computation (EC) is aimed at showing insights into the working principles of evolutionary algorithms that could be hard or even impossible to obtain through classical experimentation. These insights can be leveraged for the design of more efficient solvers, which was demonstrated, for example, in [2], which presented a genetic algorithm that achieves an asymptotically better performance than any algorithm without crossover could possibly achieve, and this even for very smooth optimization problems. Another example is the heavy-tailed alternative to the standard bit mutation suggested in [5]. This idea was shown to increase efficiency when crossing gaps in the fitness landscape. It also shows encouraging empirical performance on the recently announced *nevergrad* benchmarking platform [10].

A particularly active topic within theory of EC is *runtime analysis*, which studies the trade-offs between the number of function evaluations and the quality of the best search points that the algorithm has evaluated within this budget. Most of the works on runtime analysis are focused on bounding the expectation and the concentration of the number of fitness evaluations needed to locate a global optimum. While this seems natural from the perspective of solving a problem to optimality, finding a global optimum is infeasible in most practical applications. This has given rise to alternative indicators, which aim at making more explicit the mentioned trade-off between budget and target value. One of the prominent measures is *fixed-budget analysis*, which aims at determining which fitness to expect given a certain computational budget. Brought to the attention of the theory community by [6], a number of subsequent works [3, 4, 7, 9] proved rigorous *fixed-budget results* for some classical problems in the theory of EC.

In [1], *fixed-target analysis* was proposed as an alternative performance measure for the anytime behavior of evolutionary algorithms (EAs). Fixed-target analysis extends the classically considered optimization time by considering also first hitting times of sub-optimal target values. That is, fixed-target results make statements about the distribution of the number of function evaluations needed to find a solution that meets a minimum quality requirement.

We recall that both fixed-budget and fixed-target measures are among the standard indicators reported in empirical studies, so that the contribution of [6] and [1] is not to be seen in defining these measures, but in suggesting them for theoretical considerations. We also note that some implicit fixed-target results were already shown before the appearance of [1], see [4, 9] for examples.

Apart from suggesting fixed-target analyses, [1] also suggested to consider an alternative cost measure. Specifically, they suggest to omit function evaluations of offspring that are identical to their parents. It is argued in [1] that this rule can be assumed to be met in many applications, in particular when function evaluations are very costly. For some algorithms this non-evaluation strategy is identical to enforcing that at least one bit is flipped in each iteration, i.e., by enforcing that the offspring is different from its parent. This is the case for the well-studied (1+1) EA. Formally, the suggestion of [1] changes the (1+1) EA to a new algorithm, which is coined as "resampling" (1+1) EA in [1], or (1 + 1) $\text{EA}_{>0}$ for short.

In this paper, we analyze the fixed-target running time of the (1 + 1) $\text{EA}_{>0}$ on OneMax and BinVal. To the best of our knowledge, the number of theoretically proven fixed-target results for this algorithm is very limited. In [1] some bounds are derived for

OneMax and LeadingOnes. The bounds are parametrized by the mutation rate p, which assumed to be constant throughout the run. For the expected fixed-target time needed to solve OneMax with resampling, in [1] a quite pessimistic lower bound was proven, which we address in Section 3.1. For LeadingOnes with resampling, considering k is our target fitness value, an exact fixed-target result was obtained: $T_k = \frac{1-(1-p)^n}{2p^2}((1-p)^{1-k} - (1-p))$.

Our main contributions to the existing fixed-target results for the $(1+1)$ EA$_{>0}$ are:

- a refined upper bound for OneMax, which improves the bound from [1] by an additive term of $\Theta(n)$;
- complementary lower bounds for OneMax (we show how to leverage tools from classical optimization time analysis);
- a rather precise fixed-target bound for BinVal. Here again we can make use of existing results on the optimization time, but in a completely different way than for OneMax, as we define reductions to the same problem of smaller size.

We also highlight some issues of using classical drift analysis on the example of linear functions, which appear when using potentials that do not correlate well with the distance in the search space.

2 PRELIMINARIES

The $(1+1)$ EA$_{>0}$ is the "resampling" variant of the simple $(1+1)$ EA which requires that in each iteration at least one bit of the parent individual is mutated (whereas in the basic $(1+1)$ EA it might happen that parent and offspring are identical). The difference between parent and offspring is achieved by sampling a new offspring in case of a 0-bit flip. With this rule, the number l of bit positions to flip follows the conditional distribution $\mathrm{Bin}_{>0}(n,p)$, which assigns to each t a probability of $\Pr[l=t] = \Pr[\mathrm{Bin}(n,p)=t \mid t>0] = \binom{n}{t}p^t(1-p)^{n-t}/(1-(1-p)^n)$. Here and in the following p denotes the mutation probability. We assume without further mentioning that $0 < p \le 1/2$. The $(1+1)$ EA$_{>0}$ is outlined in Algorithm 1.

All problems considered in this work are formulated as functions $f: \{0,1\}^n \rightarrow \mathbb{R}$. We assume *maximization* as objective. Since in each subsection we consider a single problem, we will omit its explicit mention from the notation wherever possible.

By T_k we denote the expected time (measured by the number of fitness evaluations) that the $(1+1)$ EA$_{>0}$ needs to find a solution with fitness at least k. The notation $T_{i\rightarrow k}$ denotes the expected time to reach a solution with fitness at least k, when starting from a random solution of fitness equal to i. Assuming x to be a random parent of fitness $f(x)=i$ and assuming y to be its offspring, we use p_i to denote the probability that $f(y) > f(x)$ holds. We call p_i the *improvement probability* at fitness i. Furthermore, we denote by $p_{i\rightarrow k}$ the probability that $f(y)=k$ conditioned on the event of the improvement, i.e., $\Pr[f(y)=k \mid f(y)>f(x)]$. Finally, we write $x=(x_1,\ldots,x_n)$ for all $x \in \{0,1\}^n$.

Algorithm 1 $(1+1)$ EA$_{>0}$ optimizing $f: \{0;1\}^n \rightarrow \mathbb{R}$

```
p : the mutation probability
x ← UniformRandom({0;1}^n)          ▷ the best individual so far
while x is not optimized do
    Sample l ~ Bin_{>0}(n,p), the number of bit positions to flip
    y ← Mutate(x,l)                 ▷ the offspring with l flipped bits
    if f(y) ≥ f(x) then
        x ← y
    end if
end while
```

3 ONEMAX

In this section, we consider the OneMax problem, which is defined by $OM(x) = \sum_{i=1}^n x_i$. The exact expression for the probability of improvement of any solution x with $OM(x)=i$ equals

$$p_i = \sum_{j=i+1}^{n} \frac{\sum_{k=0}^{\min(i,n-j)} \binom{n-i}{k+j-i}\binom{i}{k} p^{2k+j-i}(1-p)^{n-2k-j+i}}{1-(1-p)^n}, \quad (1)$$

where we recall that the denominator comes from the condition that we enforce to flip at least one bit per iteration. Since Eq. 1 is quite complicated, we will use upper and lower bounds on p_i.

3.1 Upper Bound

We derive the upper bounds with the help of the commonly used pessimizations, that is, we assume that only one-bit flips succeed, from which it follows that, under the resampling strategy:

$$p_i \ge \frac{(n-i)\cdot p(1-p)^{n-1}}{1-(1-p)^n}$$

and the expected runtime to get from fitness i to fitness k is:

$$T_{i\rightarrow k} \le \sum_{j=i}^{k-1} p_j \le \frac{1-(1-p)^n}{p(1-p)^{n-1}} \sum_{j=i}^{k-1} \frac{1}{n-j} = \frac{1-(1-p)^n}{p(1-p)^{n-1}} \left(H_{n-i} - H_{n-k}\right),$$

where $H_j = \sum_{t=1}^{j} 1/t$ is the j-th harmonic number. With the pessimistic bound $T_k \le T_{0\rightarrow k}$ we get the result from [1]:

$$T_k \le T_{0\rightarrow k} \le \frac{1-(1-p)^n}{p(1-p)^{n-1}} \left(H_n - H_{n-k}\right). \quad (2)$$

On the other hand, the upper bound on T_k can be obtained from the distribution of the initial fitness, $\Pr[OM(x^{(0)})=i] = \binom{n}{i}2^{-n}$:

$$T_k = \sum_{i=0}^{k-1} \Pr[OM(x^{(0)})=i]\cdot T_{i\rightarrow k} \le \frac{1-(1-p)^n}{2^n p(1-p)^{n-1}} \sum_{i=0}^{k-1} \binom{n}{i} \left(H_{n-i} - H_{n-k}\right).$$

While such precision is unnecessary in proofs of the expected running time towards the optimum, as most corrections of this form influence only lower-order terms, the fixed target analysis can feel this difference for k sufficiently far away from the optimum. In particular, we are able to improve the bound (2) by roughly a $\frac{\ln 2}{p}$ margin, which is $n \ln 2$ in the case of $p = 1/n$.

Theorem 3.1. *The upper bound on the expected fixed-target runtime of the $(1+1)$ EA$_{>0}$ with the mutation probability p on OneMax of the size n and the target value $k > n/2 + \sqrt{n}\ln n$ is:*

$$T_k \le \frac{1-(1-p)^n}{p(1-p)^{n-1}} \cdot \left(H_{n/2} - H_{n-k}\right) \cdot (1-o(1)).$$

First we need to prove two lemmas about binomial coefficients.

Lemma 3.2. *For $\delta = o(1)$ the following holds:*

$$\binom{n}{\frac{n}{2}(1-\delta)} = (1 \pm o(1)) \frac{2^n}{\sqrt{\frac{\pi n}{2}}} \cdot e^{-\frac{\delta^2 n}{2}}.$$

Proof. We use the Stirling approximation for factorials:

$$\binom{n}{i} = (1 \pm o(1)) \sqrt{\frac{n}{2\pi i(n-i)}} \cdot \frac{n^n}{i^i (n-i)^{n-i}}.$$

Substitution of $(1-\delta)n/2$ for i results in the following:

$$\binom{n}{\frac{n}{2}(1-\delta)} = (1 \pm o(1)) \frac{2^n}{\sqrt{\frac{\pi n}{2}}} \cdot \frac{1}{(1-\delta^2)^{\frac{n+1}{2}}} \cdot \left(1 - \frac{2\delta}{1+\delta}\right)^{\frac{n\delta}{2}}.$$

Next we use the fact that $(1-\epsilon)^{1/\epsilon} \longrightarrow e^{-1}$ when $\epsilon \longrightarrow 0$:

$$\binom{n}{\frac{n}{2}(1-\delta)} = (1 \pm o(1)) \frac{2^n}{\sqrt{\frac{\pi n}{2}}} \cdot \frac{1}{e^{-\frac{\delta^2 (n+1)}{2}}} \cdot e^{-\frac{n\delta^2}{1+\delta}}.$$

This proves the lemma after noticing that

$$\frac{e^{\frac{-n\delta^2}{1+\delta}}}{e^{\frac{-\delta^2(n+1)}{2}}} = e^{\frac{\delta^2(n+1)}{2} - \frac{n\delta^2}{1+\delta}} = e^{-\frac{n\delta^2}{2} + \frac{\delta^2}{2} + \frac{n\delta^3}{1+\delta}} = (1 + o(1)) e^{-\frac{n\delta^2}{2}}. \qquad \square$$

Lemma 3.3. *The following inequality holds for $k < n/2$:*

$$\sum_{i=0}^{k} \binom{n}{i} \le \binom{n}{k} \cdot \frac{n-k+1}{n-2k+1} = \binom{n}{k} \cdot \left(1 + \frac{k}{n-2k+1}\right).$$

Proof. We use the facts that $\binom{n}{i-1}/\binom{n}{i} = \frac{i}{n-i+1}$ and that $\frac{i}{n-i+1} > \frac{i-1}{n-i+2}$ to bound $\binom{n}{i}$ from above as follows:

$$\binom{n}{i} \le \binom{n}{k} \cdot \left(\frac{k}{n-k+1}\right)^{k-i},$$

which enables bounding the sum in question by a sum of an infinite geometric progression to yield the desired result. $\square$

Proof of Theorem 3.1. It is enough to prove the following:

$$\sum_{i=0}^{k-1} \frac{\binom{n}{i}}{2^n} (H_{n-i} - H_{n-k}) \le (1 - o(1)) \left(H_{n/2} - H_{n-k}\right),$$

assuming $k > n/2 + \sqrt{n} \ln n$. We retain only the indices from $n/2 \pm \sqrt{n} \ln n$ on the left-hand side by showing the following:

$$\sum_{i=0}^{n/2 - \sqrt{n}\ln n} \frac{\binom{n}{i}}{2^n} \cdot H_n = o(1/n).$$

Using Lemmas 3.2, 3.3, by choosing $\delta = \frac{2\ln n}{\sqrt{n}}$, we show that

$$\sum_{i=0}^{\frac{n}{2}(1-\delta)} \frac{\binom{n}{i}}{2^n} \le (1 \pm o(1)) \cdot \frac{e^{-\frac{\delta^2 n}{2}}}{\sqrt{\frac{\pi n}{2}}} \cdot \left(1 + \frac{\frac{n}{2}(1-\delta)}{2\delta n + 1}\right)$$
$$= \frac{e^{-2(\ln n)^2}}{\Theta(\sqrt{n})} \cdot \Theta\left(\frac{\sqrt{n}}{\log n}\right) = \Theta\left(\frac{1}{n^{2\ln n} \log n}\right), \quad (3)$$

which is $o(1/n)$ even after multiplying by $H_n = O(\log n)$, so it hides in $(1 - o(1))$, and the theorem is reduced to:

$$\sum_{i=n/2-\sqrt{n}\ln n}^{n/2+\sqrt{n}\ln n} \frac{\binom{n}{i}(H_{n-i} - H_{n-k})}{2^n} \le (1 - o(1)) \left(H_{n/2} - H_{n-k}\right).$$

As $\sum_{i=n/2-\sqrt{n}\ln n}^{n/2+\sqrt{n}\ln n} \frac{\binom{n}{i}}{2^n} = 1 - o(1/n)$, which follows from (3), it remains to prove that the following expression, which bounds the change that happens when replacing H_{n-i} with $H_{n/2}$, is small:

$$\left| \sum_{i=n/2-\sqrt{n}\ln n}^{n/2+\sqrt{n}\ln n} \frac{\binom{n}{i}(H_{n-i} - H_{n/2})}{2^n} \right|.$$

We pair up the addends having matching binomial coefficients:

$$\left| \sum_{i=n/2-\sqrt{n}\ln n}^{n/2} \frac{\binom{n}{i}(H_{n-i} + H_i - 2H_{n/2})}{2^n} \right| \quad (4)$$

and note that

$$2H_{n/2} - H_{n-i} - H_i \le \sum_{j=i}^{n/2-1} \frac{1}{j} - \frac{1}{j + n/2 - i} \le \sum_{j=i}^{n/2-1} \frac{n/2 - i}{i^2}$$
$$= \left(\frac{n/2 - i}{i}\right)^2 \le \left(\frac{\sqrt{n}\ln n}{n/2}\right)^2 = O\left(\frac{(\log n)^2}{n}\right),$$

which promotes to (4) as well and completely hides into $1 - o(1)$ as $H_{n/2} - H_{n-k} = \Omega\left(\frac{\log n}{\sqrt{n}}\right)$. $\square$

3.2 Lower Bound

The main result of this section is as follows for $p = O(n^{-2/3-\varepsilon})$:

Theorem 3.4. *Let $\tilde{p} = \max(p, 1/n)$, $s_{\min} = n\tilde{p}\ln^2 n$, $s_{\max} = 1/(2\tilde{p}^2 n \ln n)$, then the lower bound on the expected fixed-target runtime of the $(1+1)$ $EA_{>0}$ with the mutation probability p on OneMax of the size n when n is big enough and the target value k is:*

$$\frac{T_k}{1 - o(1)} \ge \begin{cases} \frac{1-(1-p)^n}{p(1-p)^n} \ln \frac{1}{4\tilde{p}^3 n^2 \ln^3 n}, & \text{for } n - k \le s_{\min}; \\ \frac{1-(1-p)^n}{p(1-p)^n} \ln \frac{1}{4\tilde{p}^2 n(n-k)\ln n}, & \text{for } s_{\min} < n - k = o(s_{\max}). \end{cases}$$

Proof. Our proof is based on the proof of Theorem 6.5 from [11] and heavily uses its internals. This theorem proves the lower bound for the expected runtime of (1 + 1) EA on OneMax by observing the behaviour of the algorithm between fitness levels $n - s_{\max}$ and $n - s_{\min}$, where $s_{\min}$ and $s_{\max}$ are defined as above.

For the **first case**, the target fitness level k is above the upper fitness level $n - s_{\min}$, so T_k differs from T_n only in lower-order terms. We thus directly use the result of Theorem 6.5 [11].

For the **second case**, we replace $s_{\min}$ by $k' = n - k$ in the proof of Theorem 6.5. We also refine the choice of the parameter β, that controls the size and the probability of tolerable large jumps, from $\beta = 1/\ln n$ to $\beta = b/k'$, where $b = \tilde{p} n \ln n$ as in Theorem 6.5, which still satisfies the conditions of [11, Theorem 2.2] used to derive the runtime bounds. The factor $(1-\beta)/(1+\beta)$ would still be $1 - o(1)$ with this choice. The replacement of $s_{\min}$ by $k' = n - k$ affects the logarithm of the ratio of potentials, which now becomes $\ln(s_0/k') = \ln(1/(4\tilde{p}^2 n \ln nk'))$. $\square$

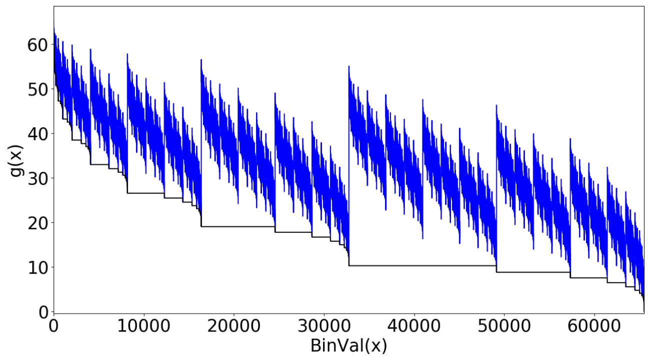

Figure 1: Plot of the potential function γ_i applied to BinVal, $p = 1/n, \alpha = \ln\ln n, n = 16$

4 NOTES ON LINEAR FUNCTIONS

Can the existing tools for runtime analysis be used to obtain good fixed-target results? Some of them have been successfully used for problems like OneMax and LeadingOnes, but for wider function classes (i.e., linear functions $f(x) = \sum_{i=1}^{n} w_i x_i$, where the w_i are constant weights) the answer is not straightforward. We consider a linear function with weights equal to successive powers of two, called BinVal:

$$\text{BinVal}(x) = \sum_{i=1}^{n} 2^{i-1} \cdot x_i.$$

This function can be seen as an "extreme" linear function: a direct application of the multiplicative drift theorem [8, Theorem 11] yields an $O(n^2)$ running time for the (1 + 1) EA with the mutation probability $p = 1/n$, not the true $O(n \log n)$ bound. A typical workaround is to apply drift theorems to some other functions ("potentials") that possibly depend on the individuals. For instance, Theorem 4.1 from [11], which proves upper bounds for (1+1) EA on linear functions, uses the following weights instead of the original weights w_i, assuming $w_i \le w_{i+1}$ and $g_1 = 1$:

$$g_i = \min\left\{ g_{i-1} \cdot \frac{w_i}{w_{i-1}}, \left(1 + \frac{\alpha p}{(1-p)^{n-1}}\right)^{i-1} \right\}.$$

This idea yields an upper bound of $(1 + o(1))(e^c/c)n \ln n$ on the expected running time of (1 + 1) EA on all linear functions with mutation probability $p = c/n$ by choosing $\alpha = \ln\ln n$. To apply it to the fixed-target results, we need to choose a new lower bound $s_{\min}$ to be the minimum potential of the all individuals with $f(x) < k$, where k is our target fitness. The plot of the potential value as a function of the fitness, for BinVal with $n = 16$ and $\alpha = \ln\ln n$, is given in Fig. 1. The lower envelope of the blue curve is the value for $s_{\min}$ to take assuming x is the target fitness. Fig. 1 shows that the upper bounds following from Theorem 4.1 would be very imprecise. However, for BinVal we may apply the existing tools in a completely different way to get quite sharp bounds.

Theorem 4.1. *The expected fixed-target runtime of* (1+1) $EA_{>0}$ *on* BinVal *with the problem size n, the target fitness k and the mutation probability* $p = O(n^{-2/3-\varepsilon})$ *satisfies:*

$$T_k \ge (1 - o(1)) \frac{1 - (1-p)^n}{p(1-p)^{n^-}} \min\left\{ \ln n^-, \ln \frac{1}{p^3 (n^-)^2} \right\},$$

$$T_k \le \frac{pn^+ \alpha^2 (1-p)^{1-n^+} + \alpha \left(\ln \frac{1}{p} + (n^+ - 1)\ln(1-p) + 1 \right)}{(1-p)^{n^+-1} \cdot p(\alpha - 1) \cdot (1 - (1-p)^n)^{-1}},$$

where $n^- = n - \lceil \log_2(2^n - k) \rceil$ *and* $n^+ = n - \lfloor \log_2(2^n - k) \rfloor$, *and* α *can be chosen appropriately, even depending on n or* n^+.

Proof. Note that n^- and n^+ are chosen in such a way that, assuming x^- is an integer written in binary notation with n^- leading ones and $n - n^-$ following zeros, and x^+ similarly depends on n^+, it holds that $x^- \le k \le x^+$ and $n^- + 1 \ge n^+$. That is, in order to reach the target fitness k, it is necessary to guess right n^- most significant bits, and it is sufficient to guess right n^+ most significant bits. This effectively reduces the problem to solving a smaller BinVal problem completely, assuming the same mutation rate.

The lower bound then follows directly from [11, Theorem 6.5], and the upper bound follows directly from [11, Theorem 4.1], while noting that we use (1 + 1) $EA_{>0}$, which introduces a factor of $(1 - (1-p)^n)$ to both bounds, where n is unchanged. □

5 CONCLUSION

We improved the existing fixed-target results for (1 + 1) $EA_{>0}$ on OneMax and expanded them with lower bounds. For linear functions, it was shown that traditional drift analysis is hard to apply. We left the further analysis of this general problem for the future work. However, a rather precise bound was obtained for the particular case of the BinVal problem.

ACKNOWLEDGMENTS

Our work was supported by the Government of Russian Federation (Grant 08-08) and by a public grant as part of the Investissement d'avenir project, reference ANR-11-LABX-0056-LMH, LabEx LMH.

REFERENCES

[1] Eduardo Carvalho Pinto and Carola Doerr. 2018. Towards a More Practice-Aware Runtime Analysis of Evolutionary Algorithms. http://arxiv.org/abs/1812.00493 A short abstract appeared in the proceedings of *Artificial Evolution* conference (EA'17).

[2] Benjamin Doerr, Carola Doerr, and Franziska Ebel. 2015. From black-box complexity to designing new genetic algorithms. *Theoretical Computer Science* 567 (2015), 87–104.

[3] Benjamin Doerr, Carola Doerr, and Jing Yang. 2016. Optimal Parameter Choices via Precise Black-Box Analysis. In *Proceedings of Genetic and Evolutionary Computation Conference.* ACM, 1123–1130.

[4] Benjamin Doerr, Thomas Jansen, Carsten Witt, and Christine Zarges. 2013. A method to derive fixed budget results from expected optimisation times. In *Proceedings of Genetic and Evolutionary Computation Conference.* ACM, 1581–1588.

[5] Benjamin Doerr, Huu Phuoc Le, Régis Makhmara, and Ta Duy Nguyen. 2017. Fast genetic algorithms. In *Proceedings of Genetic and Evolutionary Computation Conference.* Full version available at http://arxiv.org/abs/1703.03334.

[6] Thomas Jansen and Christine Zarges. 2014. Performance analysis of randomised search heuristics operating with a fixed budget. *Theoretical Computer Science* 545 (2014), 39–58.

[7] Thomas Jansen and Christine Zarges. 2014. Reevaluating Immune-Inspired Hypermutations Using the Fixed Budget Perspective. *IEEE Transactions on Evolutionary Computation* 18 (2014), 674–688.

[8] Johannes Lengler. 2017. Drift Analysis. http://arxiv.org/abs/1712.00964

[9] Johannes Lengler and Nicholas Spooner. 2015. Fixed budget performance of the (1+1) EA on linear functions. In *Foundations of Genetic Algorithms XIII.* ACM, 52–61.

[10] Jérémy Rapin and Olivier Teytaud. 2018. Nevergrad - A gradient-free optimization platform. https://GitHub.com/FacebookResearch/Nevergrad

[11] Carsten Witt. 2013. Tight bounds on the optimization time of a randomized search heuristic on linear functions. *Combinatorics, Probability and Computing* 22, 2 (2013), 294–318.