Maximizing Drift is Not Optimal for Solving OneMax

Nathan Buskulic
Sorbonne Université, Paris, France

Carola Doerr
Sorbonne Université, CNRS, LIP6, Paris, France

ABSTRACT

It seems very intuitive that for the maximization of the OneMax problem $f(x) := \sum_{i=1}^{n} x_i$ the best that an elitist unary unbiased search algorithm can do is to store a best so far solution, and to modify it with the operator that yields the best possible expected progress in function value. This assumption has been implicitly used in several empirical works. [Doerr, Doerr, Yang: GECCO 2016] formally proved that this approach is indeed almost optimal.

In this work we demonstrate that drift maximization is *not* optimal. More precisely, we show that for most fitness levels between $n/2$ and $2n/3$ the optimal mutation strengths are larger than the drift-maximizing ones. This implies that the optimal RLS is more risk-affine than the variant maximizing the step-wise expected progress. We show similar results for the mutation rates of the classic (1+1) Evolutionary Algorithm (EA) and its resampling variant, the (1+1) $\text{EA}_{>0}$.

As a result of independent interest we show that the optimal mutation strengths, unlike the drift-maximizing ones, can be even.

CCS CONCEPTS

• **Theory of computation → Random search heuristics**;

1 INTRODUCTION

It is well understood that iterative optimization heuristics like local search variants, evolutionary algorithms, estimation of distribution algorithms, etc. can benefit from non-static choices of the parameters that determine their search radius, population size, or selective pressure. The question *how* to select these parameters dynamically is the subject of *parameter control,* which studies different techniques to achieve a good fit between suggested and optimal parameter values.

Complementing a diverse body of empirical works demonstrating advantages of parameter control mechanisms [8], there is an increasing interest in proving such benefits by mathematical means [4]. The vast majority of theoretical works consider the optimization of OneMax, the problem of maximizing the function $\text{Om} : \{0,1\}^n \to \mathbb{R}, x \mapsto \sum_{i=1}^{n} x_i$. OneMax also plays a prominent role in empirical research on parameter control. In both communities, it is argued that the consideration of OneMax provides a "sterile EC-like environment" [6], in which the optimal parameter values are well understood.

GECCO '19 Companion, July 13–17, 2019, Prague, Czech Republic

ACM ISBN 978-1-4503-6748-6/19/07…$15.00
https://doi.org/10.1145/3319619.3321952

In light of the existing literature it is interesting to note that most works, implicitly or explicitly, assume that for the considered algorithms the optimal strategy for the maximization of OneMax is a greedy selection of the best so far solution, and the variation of the same by the mutation rate/step size that maximizes the expected gain in function value, see, for example, [1, 7]. Thierens [9] explicitly argues that a particularly useful property of OneMax, which makes this problem a very suitable benchmark for adaptive operator selection, is the fact that the reward of an operator can be computed exactly. He then proceeds by comparing the step-wise expected fitness gains made by different operators, and ranks operators by this value. He thus uses as underlying assumption that drift-maximization is optimal.

That this widely believed-to-be-optimal *drift-maximizing* strategy is indeed almost optimal was formally proven in [5]. More precisely, it is shown in [5] that the best unary unbiased black-box algorithm for OneMax cannot be better by more than an additive $o(n)$ term than the Randomized Local Search (RLS) variant that flips in each iteration the drift-maximizing number of bits in a best-so-far solution. Both algorithms have an expected optimization time $n \ln(n) - cn \pm o(n)$, for a constant c between 0.2539 and 0.2665.

We demonstrate in this work that maximizing drift is *not* optimal neither for RLS nor for the (1+1) EA nor for its resampling variant, the $(1+1)$ $\text{EA}_{>0}$, suggested in [3].

In the full version of our work, which is available in [2], we explain where the difference between optimal and drift-maximizing strategies comes from, define precisely how to obtain the optimal mutation rates, numerically compute these for some selected dimensions up to $n = 10,000$, and analyze the differences between drift-maximizing and optimal mutation rates. We also compare the performances of optimal and drift-maximizing algorithms, and show that the differences in mutation rates/step sizes – albeit significant – result only in marginal differences in terms of overall running time. Given the above-mentioned results in [5], the last statement is not surprising. The main contribution of our work is therefore not to be found in tremendous performance gains, but in new structural insights for the optimization of OneMax, the arguably most widely used benchmark for parameter control and adaptive operator selection mechanisms.

We note that the argument *why* drift-maximization is not optimal is quite easy to understand. Basically, our result is build upon the observation that the drift-maximizer values a potential fitness progress of i by exactly this gain. More precisely, in the computation of the expected drift the probability of creating an offspring y of x is multiplied by the difference $\max\{0, \text{Om}(y) - \text{Om}(x)\}$, for each possible offspring y. The optimal algorithms, however, value a fitness gain of i by the gain in the expected remaining running time. Since this difference in expected remaining running time is much larger than the fitness difference, the optimal RLS and $(1+1)$ EA variants use mutation rates that are larger than the drift-maximizing ones. Put differently, they trade a smaller expected progress for a slightly

larger probability of making a larger fitness gain. That is, the optimal algorithms are more risk-affine than the drift-maximizing ones. This quite intuitive fact seems to have been overlooked in the evolutionary computation community.

Full Version and Online Repository. The full version of this work can be found in [2]. The GitHub page of this project provides drift-maximizing and optimal step sizes/mutation rates for problem dimensions up to $n = 10{,}000$ and for the three algorithms RLS, the (1 + 1) EA, and the (1 + 1) $\text{EA}_{>0}$. Detailed performance data are available for 16 different variants of these algorithms.

2 EXAMPLE: RLS

We briefly sketch the main effects for RLS.

We denote by RLS_{opt} the RLS variant which flips in each iteration exactly $k_{\text{opt}}(\text{OM}(x))$ pairwise different bits which are chosen uniformly at random. Here $k_{\text{opt}} : [0..n-1] \to [0..n]$ is a function which maps each fitness value to a mutation strength such that the overall expected running time is minimized. Likewise, $\text{RLS}_{\text{drift}}$ is the RLS variant which uses the drift-maximizing function drift to determine the mutation strengths. Put differently, k_{drift} is the function which maps each fitness level ℓ to the mutation strength k that maximizes the expected progress

$$\begin{aligned}&\mathbb{E}[\Delta(n,\ell,k)] := \qquad (1)\\ &\mathbb{E}[\max\{\text{OM}(y)-\text{OM}(x),0\} \mid \text{OM}(x)=\ell, y \leftarrow \text{flip}_k(x)]\\ &\sum_{i=\ell+1}^{\ell+k} (i-\ell)\mathbb{P}[\text{OM}(y)=i \mid \text{OM}(x)=\ell, y \leftarrow \text{flip}_k(x)]\\ &= \sum_{i=\lceil k/2\rceil}^{k} \frac{\binom{n-\ell}{i}\binom{\ell}{k-i}(2i-k)}{\binom{n}{k}}.\end{aligned}$$

Interestingly, it suffices to regard $n = 3$ for an example for which $\text{RLS}_{\text{opt}} \neq \text{RLS}_{\text{drift}}$. The following table summarizes for $n = 3$ the functions k_{drift}, k_{opt}, and the expected remaining running times $\mathbb{E}[T_{\text{drift}}(\ell)]$ and $\mathbb{E}[T_{\text{opt}}(\ell)]$ for $\text{RLS}_{\text{drift}}$ and RLS_{opt}, respectively, when starting in a solution x of fitness $\text{OM}(x) = \ell$. In column $p^0(\ell)$ we list the probability that a random initial solution has fitness value ℓ. Since uniform random initialization is used, $p^0(\ell) = \binom{n}{\ell}/2^n$. The last line provides the overall expected optimization time of both algorithms. Note that, by the law of total probability, $\mathbb{E}[T] = 1 + \sum_{i=0}^{n} p^0(\ell)\mathbb{E}[T(\ell)]$.

ℓ	$p^0(\ell)$	$k_{\text{drift}}(\ell)$	$\mathbb{E}[T_{\text{drift}}(\ell)]$	$k_{\text{opt}}(\ell)$	$\mathbb{E}[T_{\text{opt}}(\ell)]$
3	1/8	-	0	-	0
2	3/8	1	3	1	3
1	3/8	3	4	2	3
0	1/8	3	1	3	1
$\mathbb{E}[T]$		3.75		3.375	

Optimal Mutation Strengths Need Not be Uneven. With this example, we not only prove that $\text{RLS}_{\text{opt}} \neq \text{RLS}_{\text{drift}}$, but we also make another interesting observation, which concerns the parity of the values $k_{\text{opt}}(n,\ell)$. It was proven in [5] that k_{drift} takes only odd values, since for every k the expected drift of flipping $2k$ bits is strictly smaller than that of flipping $2k+1$ bits. The example above shows that the situation is different for k_{opt}. More precisely, we have seen that in the situation $n = 3$ and $\ell = 1$ flipping 2 bits is optimal.

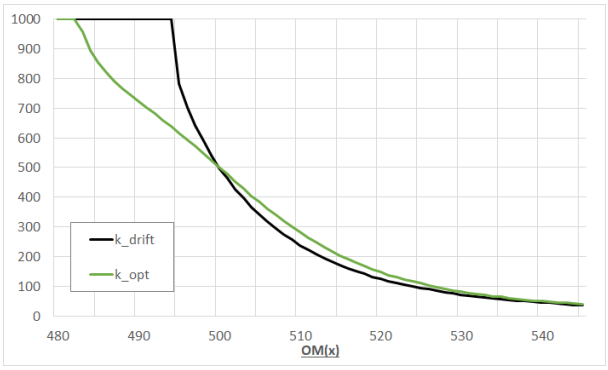

Figure 1: Comparison of $k_{\text{opt}}(n,\ell)$ and $k_{\text{drift}}(n,\ell)$ for $n = 1{,}000$ (zoom into fitness levels $480 \le \ell \le 545$).

Figure 1 plots the interesting region of $k_{\text{opt}}(\ell)$ and $k_{\text{drift}}(\ell)$ for $n = 1{,}000$; this picture is very similar across all dimensions n. In particular it holds for all n that the curves cross at fitness level $\ell = n/2$. For smaller values, the optimal mutation strengths are smaller or identical to drift-maximizing ones, and the situation is reversed for fitness levels $\ell > n/2$.

Acknowledgments. We thank Thomas Bäck for several valuable discussions on the history of adaptive parameter settings.

Our research is supported by the *Paris Ile-de-France Region*, by a public grant as part of the Investissement d'avenir project, reference *ANR-11-LABX-0056-LMH*, LabEx LMH, and by *COST Action CA15140*.

REFERENCES

[1] Thomas Bäck. 1993. Optimal Mutation Rates in Genetic Search. In *Proc. of the 5th International Conference on Genetic Algorithms (ICGA'93)*. Morgan Kaufmann, 2–8.

[2] Nathan Buskulic and Carola Doerr. 2019. Maximizing Drift is Not Optimal for Solving OneMax. *CoRR* abs/1904.07818 (2019). arXiv:1904.07818 http://arxiv.org/abs/1904.07818 See also https://github.com/NathanBuskulic/OneMaxOptimal for more project data.

[3] Eduardo Carvalho Pinto and Carola Doerr. 2018. Towards a More Practice-Aware Runtime Analysis of Evolutionary Algorithms. *CoRR* abs/1812.00493 (2018). arXiv:1812.00493 http://arxiv.org/abs/1812.00493 An extended abstract appeared in *Proc. of Artificial Evolution (EA'17)*, pages 298–305.

[4] Benjamin Doerr and Carola Doerr. 2018. Theory of Parameter Control Mechanisms for Discrete Black-Box Optimization: Provable Performance Gains Through Dynamic Parameter Choices. In *Theory of Randomized Search Heuristics in Discrete Search Spaces*, Benjamin Doerr and Frank Neumann (Eds.). Springer. To appear. Available online at https://arxiv.org/abs/1804.05650.

[5] Benjamin Doerr, Carola Doerr, and Jing Yang. 2016. Optimal Parameter Choices via Precise Black-Box Analysis. In *Proc. of Genetic and Evolutionary Computation Conference (GECCO'16)*. ACM, 1123–1130.

[6] Álvaro Fialho, Luís Da Costa, Marc Schoenauer, and Michèle Sebag. 2008. Extreme Value Based Adaptive Operator Selection. In *Proc. of Parallel Problem Solving from Nature (PPSN'08) (Lecture Notes in Computer Science)*, Vol. 5199. Springer, 175–184.

[7] Álvaro Fialho, Luís Da Costa, Marc Schoenauer, and Michèle Sebag. 2009. Dynamic Multi-Armed Bandits and Extreme Value-Based Rewards for Adaptive Operator Selection in Evolutionary Algorithms. In *Proc. of Learning and Intelligent Optimization (LION'09) (Lecture Notes in Computer Science)*, Vol. 5851. Springer, 176–190.

[8] Giorgos Karafotias, Mark Hoogendoorn, and A.E. Eiben. 2015. Parameter Control in Evolutionary Algorithms: Trends and Challenges. *IEEE Transactions on Evolutionary Computation* 19 (2015), 167–187.

[9] Dirk Thierens. 2009. On benchmark properties for adaptive operator selection. In *Proc. of Genetic and Evolutionary Computation Conference (GECCO'09), Companion Material*. ACM, 2217–2218.