Illustrating the Trade-Off between Time, Quality, and Success Probability in Heuristic Search

A Discussion Paper

Ivan Ignashov
ITMO University
Saint Petersburg, Russia
i.ignashov@yandex.ru

Arina Buzdalova
ITMO University
Saint Petersburg, Russia
abuzdalova@gmail.com

Maxim Buzdalov
ITMO University
Saint Petersburg, Russia
mbuzdalov@gmail.com

Carola Doerr
Sorbonne Université, CNRS
Paris, France
Carola.Doerr@lip6.fr

ABSTRACT

Benchmarking aims to investigate the performance of one or several algorithms for a set of reference problems by empirical means. An important motivation for benchmarking is the generation of insight that can be leveraged for designing more efficient solvers, for selecting a best algorithm, and/or for choosing a suitable instantiation of a parametrized algorithm. An important component of benchmarking is its design of experiment (DoE), which comprises the selection of the problems, the algorithms, the computational budget, etc., but also the performance indicators by which the data is evaluated. The DoE very strongly depends on the question that the user aims to answer. Flexible benchmarking environments that can easily adopt to users' needs are therefore in high demand.

With the objective to provide such a flexible benchmarking environment, the recently released IOHprofiler not only allows the user to choose the sets of benchmark problems and reference algorithms, but provides in addition a highly interactive, versatile performance evaluation. However, it still lacks a few important performance indicators that are relevant to practitioners and/or theoreticians. In this discussion paper we focus on one particular aspect, the probability that a considered algorithm reaches a certain target value within a given time budget. We thereby suggest to extend the classically regarded fixed-target and fixed-budget analyses by a fixed-probability measure. Fixed-probability curves are estimated using Pareto layers of (target, budget) pairs that can be realized by the algorithm with the required certainty. We also provide a first implementation towards a fixed-probability module within the IOHprofiler environment.

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

ACM ISBN 978-1-4503-6748-6/19/07…$15.00
https://doi.org/10.1145/3319619.3326895

CCS CONCEPTS

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

KEYWORDS

Black-Box Optimization, Benchmarking, Iterative Optimization Heuristics, Performance Evaluation

ACM Reference Format:
Ivan Ignashov, Arina Buzdalova, Maxim Buzdalov, and Carola Doerr. 2019. Illustrating the Trade-Off between Time, Quality, and Success Probability in Heuristic Search: A Discussion Paper. In *Genetic and Evolutionary Computation Conference Companion (GECCO '19 Companion), July 13–17, 2019, Prague, Czech Republic.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3319619.3326895

1 INTRODUCTION

Search heuristics are general-purpose optimization algorithms which aim to identify high-quality solutions by an iterative evaluation of solution candidates. The function values of these *search points* are used to adjust the strategy by which the next round of samples are generated. Search heuristic are thus in sharp contrast with constructive heuristics, which aim at generating suitable solutions by analyzing and manipulating the problem data in a direct fashion. Among the most prominent search heuristics are local search algorithms (such as first ascent, steepest ascent, Simulated Annealing, Threshold Accepting, etc.), genetic and evolutionary algorithms, estimation of distribution algorithms, ant colony optimization algorithms, artificial immune systems, as well as meta-concepts (e.g., so-called "hyper-heuristics") that alternate between different strategies during the optimization process.

A key challenge in applying search heuristics in practice is the selection of a most suitable algorithm for the given problem at hand. What is more is that most search heuristics are parametrized, and ask the user to specify the search radius, the number of samples that are to be evaluated in each iteration, and the size of the allocated memory – to name but a few parameters of a typical search heuristic. Both the *algorithm selection* but also the latter-mentioned *algorithm configuration* problem require a very solid experience with heuristics and/or excessive empirical data from which performance extrapolation models can be build. Training human experts

and computer programs for these tasks is the purpose of *benchmarking*. Put differently, benchmarking aims to provide data-driven evidence to favor certain algorithms and/or instantiations over others.

1.1 Performance Criteria

As part of a carefully crafted design of experiments, a meaningful benchmarking effort also comprises the task of choosing the performance measure by which the algorithms are to be evaluated. This is a very tricky part, since we often observe that a heuristic that is particularly good for small computational budgets is outperformed by other heuristics for budgets of medium or large size (and vice versa). We also observe that an algorithm behaving well *on average* need not necessarily show very *stable* performance. Which algorithm to favor above others depends very strongly on the goal of the training effort:

- Users with a predefined computational budget will care mostly about the *fixed-budget view*, which answer questions of the type *What is the best/median/average/... quality of the solutions that one can expect to obtain with Algorithm A within a given computational budget B?* On the other hand, users requiring solutions of a certain quality threshold are likely to focus on the *fixed-target view* instead, by asking questions of the type *How long do we need to wait (on average/in the worst case/in the best case/...) until Algorithm A has identified a solution of quality at least φ?*
- In applications where failures can cause substantial negative effects (e.g., high cost), we require a very high probability that the selected algorithm is capable of finding a solution of desired quality. If, on the other hand, we plan to run the same algorithm on several parallel machines, we can typically accept a much smaller success probability.

These considerations show that the performance evaluation of search heuristics has three main criteria:

(1) solution quality,
(2) computational budget, and
(3) probability of success.

In this light, algorithm selection and configuration classify as multi-objective optimization problems.

As usual in multi-objective optimization [1, 2], an important object of a multi-criteria optimization problem is the Pareto front, which describes the best trade-offs that one can achieve without sacrificing at least one of the objectives. Illustrating three-dimensional Pareto fronts is non-obvious. The two most common ways to display such Pareto fronts is by fixing one of the three objectives and showing the two-dimensional section of the Pareto front, which is simply a curve in a two-dimensional coordinate system. By repeatedly fixing the first objective to different values, a number of sections can be shown at the same time. A similar approach is used in evolutionary computation, where the probably most frequently used performance charts plot the median solution quality of an algorithm against the allocated time budget (*fixed-budget plots*), thus effectively fixing the success probability to 50%. Similarly common are *fixed-target plots*, which illustrate the median first-hitting times of an algorithm against the target values, thus again fixing the success probability to 50%, but inverting the dependence of time and quality.

While it is also common to add to these curves some confidence intervals (for example, in the form of box-plots) it is much less common to explicitly show the time-quality trade-off for quantiles (i.e., success probabilities) different from 50%. A notable exception to the median-success plots are empirical cumulative distribution function (ECDF) curves [6], which aggregate success-probabilities for a *set* of fixed targets. In the evolutionary computation literature, ECDF curves are particularly common in the assessment of continuous black-box optimization algorithms. At the same time, they seem to play only a marginal role in the assessment of discrete optimization techniques. A key driver for the success of ECDF curves in the continuous domain can be seen in the fact that they are part of the standard performance reports offered by the COCO platform [7], which is an important benchmarking environment for continuous black-box optimization heuristics.

1.2 Motivation for Our Work

The performance reports offered by the COCO environment are rather static, in that there is no immediate interface for the user to chose the target values for which the ECDF curves are to be generated. This is in sharp contrast to the philosophy behind the IOHprofiler – a recently released benchmarking software which primarily targets discrete optimization benchmarking. IOHprofiler [5], or, more precisely, its post-processing part, the IOHanalyzer, allows the user to choose for which target values the algorithms are to be assessed. These targets can be chosen for each problem individually. Apart from ECDF curves, which – as mentioned above – are much less commonly used in discrete optimization, IOHprofiler offers a number of different fixed-target and fixed-budget analyses and plots. However, following the common trend, a *fixed-probability* section of the performance evaluation is currently missing – a gap that we address in this current work, and that we start to fill by proposing a first step towards a meaningful three-dimensional performance evaluation.

The motivation for our work lies in applications where a decent success probability needs to be ensured. Put differently, we are concerned with use cases in which a reliable performance is required. In such cases, one wishes to fix the maximum failure probability $\varepsilon > 0$ that the user is willing to tolerate, and we are interested in understanding how the quality that can be ensured with probability $1 - \varepsilon$ evolves with the budget, or how much time is required to achieve a given target precision with failure probability at most ε.

1.3 Availability of Our Implementation

The IOHprofiler modules described in this work are available at http://iohprofiler.liacs.nl/dev/. The user can load there some reference benchmark data, and can upload her/his own data to test the modules. Since the work described in this workshop paper is of preliminary nature, several modifications can be expected for the next months. Readers interested in contributing to this effort are invited to get in touch.

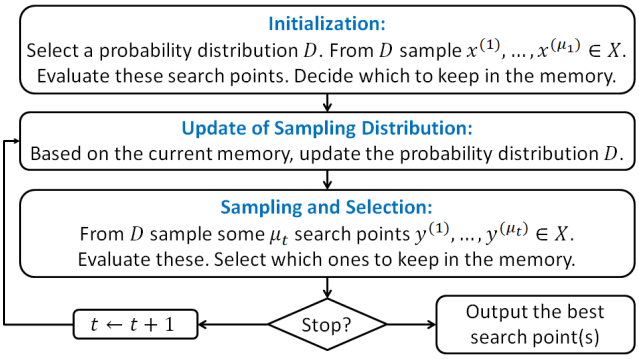

Figure 1: Illustration of iterative search heuristics.

1.4 Related Work

We point out that neither our approach nor our performance measure are new. In essence, we reduce the performance evaluation of iterative search heuristics to a multi-objective optimization problem with three different criteria: time, quality, success probability. Multi-objective optimization is an important sub-domain in itself, and forms a core discipline within the evolutionary computation community. Many different ways to evaluate the performance of search heuristics for multi-criteria optimization problems are discussed in the literature, see [11] for a critical survey. A measure closely related to our fixed-probability curves are the *empirical attainment surfaces*, for which a discussion of computational aspects can be found in [8].

2 THE TRADE-OFF BETWEEN TIME, QUALITY, AND SUCCESS PROBABILITY

Before drawing the reader's attention to the fixed-probability performance measure, we briefly summarize in this section a few basic performance indicators. As mentioned above, the most commonly reported performance measures in the search heuristics literature are *fixed-target running times*, *fixed-budget results*, and *ECDF curves*. Our focus in this work is on the evaluation of single benchmark functions; i.e., we do not address here the question how to aggregate performances across different benchmark problems. We therefore disregard ECDF curves in the following descriptions.

Our focus is on iterative search heuristics, as illustrated in Figure 1.

2.1 Fixed-Probability Measures

We formulate all measures for *maximization problems*, but all adjustments required in the case of minimization problems are straightforward. For all positive integers k we abbreviate $[k] := \{1, 2, \ldots, k\}$ and set $[0..k] := [k] \cup \{0\}$.

Definition 2.1 (Fixed-Target, Fixed-Budget, and Fixed-Probability Performance Measures). Let A be an algorithm that is run r times on a maximization problem $f : X \to \mathbb{R}$, with a maximum budget of $B \in \mathbb{N}$ function evaluations. For each run $i \in [r]$ and each value $t \in [B]$ let $x^{(i,t)}$ be the t-th search point queried by the algorithm (breaking ties randomly in case of parallel evaluations). Finally, for each $t \in [B]$, let X^t be the random distribution over X that algorithm A uses to sample the t-th search point.

For a given target value $\varphi \in \Phi := \{f(x) \mid x \in X\}$, the most relevant quantities in the fixed-target analysis are as follows.

- $S(A, f, \varphi) := \{j \in [r] \mid \exists t \in [B] : f(x^{(j,t)}) \geq \varphi\}$ is the *set of successful runs*, $s(A, f, \varphi) := |S(A, f, \varphi)|$ the *number of successful runs*, and $sp(A, f, \varphi) := s(A, f, \varphi)/r$ the *success probability*.[1]
- $T(A, f, i, \varphi) := \min\{t \mid f(x^{(i,t)}) \geq \varphi\}$ is the *first hitting time* of target value φ in run i,
- $\mathrm{AHT}(A, f, \varphi) := \mathrm{avg}\{T(A, f, j, \varphi) \mid j \in [S(A, f, \varphi)]\}$ is the *average first hitting time* (AHT) of algorithm A for target value φ on function f.
- When at least one run was successful for target φ, we call $\mathrm{ERT}(A, f, \varphi) := (1 - sp(A, f, \varphi))B + \mathrm{avg}\{T(A, f, j, \varphi) \mid j \in [S(A, f, \varphi)]\}$ the *expected running time* (ERT).[2] When $S(A, f, \varphi) = \emptyset$, we set $\mathrm{ERT}(A, f, \varphi) = \infty$. Note here that the ERT measure implicitly assumes an algorithm that is restarted when no point of sufficient quality has been found within the given budget – an assumption that is not always met in practice. It is therefore recommended to carefully check whether the use of AHT or ERT values is more appropriate.
- Instead of looking at averages, it is also common to look at the median and other quantiles. For each value $0 \leq q \leq 1$ with $s(A, f, \varphi) \geq qr$ we set $T(A, f, \varphi, q)$ equal to

  $$\min\{t \mid |\{j \in [S(A, f, \varphi)] \mid T(A, f, j, \varphi) \leq t\}| \geq qr\},$$

  and we set $T(A, f, \varphi, q) := B$ otherwise. This defines the empirical q-th quantile running time of all runs for target value φ.
  $T(A, f, \varphi, q)$ approximates $T^*(A, f, \varphi, q) := \min\{b \in [B] \mid \Pr[\max\{f(X^t) \mid t \in [b]\} \geq \varphi] \geq q\}$, the fixed-probability budget required to obtain a solution of objective value at least φ with probability at least q. In the following we speak of $T(A, f, \varphi, q)$ as the estimated fixed-probability budget evaluated at target φ.

These fixed-target measures have their complementary *fixed-budget* performance indicators. For a run $i \in [r]$ and a budget $b \in [B]$,

- $V(A, f, i, b) := \max\{f(x^{(i,t)}) \mid t \in [b]\}$ denotes the *current-best fitness value* or *best target (value)* at time/budget b, and
- $V(A, f, b) := \mathrm{avg}\{V(A, f, j, b) \mid j \in [r]\}$ is the average best target value for this budget b.
- For each $0 \leq q \leq 1$ the q-th quantile target value $V^*(A, f, b, q)$ at budget b is

  $$\max\{\varphi \in \Phi \mid \Pr[\max\{f(X^t) \mid t \in [b]\} \geq \varphi] \geq q\},$$

  the largest value φ for which the probability that the best of the first b evaluated solution candidates has fitness φ or

[1] Note here that it would be more accurate to speak of the "estimated success probability" but since all quantities studied in this work are based on empirical data, we omit the explicit mentioning of the term "estimated" – here and throughout this work.
[2] Note that this name, despite its common use, is potentially misleading, since the term "expectation" suggests that this is a mathematical quantity, whereas it reports merely empirical averages.

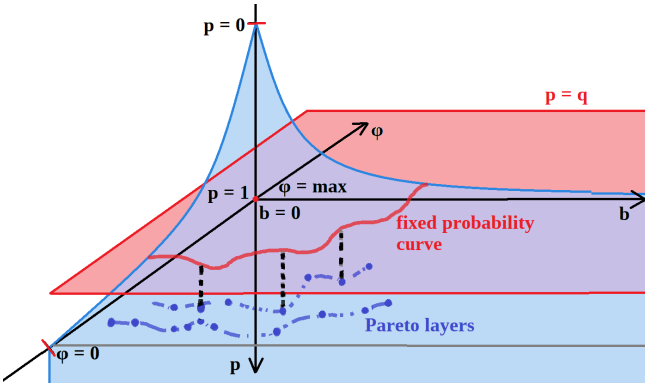

Figure 2: An example (φ, b, p) manifold (blue) induced by algorithm A and problem f, and a fixed probability plane (red). The fixed-probability curve is defined by $V^*(A, f, \cdot, q)$ and $T^*(A, f, \cdot, q)$.

larger is at least q. In practice we approximate $V^*(A, f, b, q)$ by $V(A, f, b, q)$, which is defined as

$$\max\{\varphi \in \Phi \mid |\{j \in [r] \mid V(A, f, j, b) \geq \varphi\}| \geq qr\}.$$

To be very clear, we formulate the fixed-probability measures more explicitly. For each probability $0 < q < 1$ the fixed-probability measures for success probability q are the functions

- $V^*(A, f, \cdot, q) : [B] \to \mathbb{R}, b \mapsto V^*(A, f, b, q)$, and
- $T^*(A, f, \cdot, q) : \Phi \to \mathbb{R}, \varphi \mapsto T^*(A, f, \varphi, q)$.

We note that these functions define two-dimensional fixed-probability curves (see Fig. 2), an aspect that we will discuss in more detail below.

Our goal is to add to the IOHprofiler environment, which currently covers the fixed-target and the fixed-budget perspective, a fixed-probability section, which automates the computation and illustration of the fixed-probability measures defined above.

2.2 Intensity Plots

In this section we describe the intensity plots which we implemented in the IOHprofiler. The intensity plots are one way of illustrating the fixed-probability measures. Examples for the intensity plots will be provided in Section 3.

The intensity plots are implemented within both fixed-target and fixed-budget plots, which means that they are constructed in the two-dimensional space comprised of the time budget b and the target value φ. The third dimension (success probability p) is illustrated by the intensity of the plot's color. The success probability is evaluated empirically based on the number of runs which satisfy the corresponding target and budget constraints.

Below we interpret the definitions of fixed-probability measures given in Section 2.1 and show how the implemented intensity plots relate to them. Imagine a three-dimensional space comprised of function value φ, time budget b and success probability p illustrated in Fig. 2. All the (φ, b, p) triplets possible for given A, f form a manifold. When we fix some probability q, we observe a projection of this manifold in a two-dimensional space of varying φ and b. The fixed-probability functions $V^*(A, f, \cdot, q)$ and $T^*(A, f, \cdot, q)$ describe this projection, which is a fixed-probability curve.

On the intensity plot, we also consider a two-dimensional space of φ and b, and observe some empirically obtained points $(f(x^{(i,t)}), t)$ in this space. The traces of this points approximate the above mentioned curves. We then consider Pareto layers to estimate the fixed probabilities q which generate these curves. In other words, the Pareto layers may be seen as the approximations of the fixed-probability curves.

2.3 Construction of the Pareto Layers

To formally describe the intensity plot, consider a two-criterion problem of maximizing the target value φ and minimizing the time budget b. Thus we have the problem of finding the Pareto front of (φ, b) points in the $\mathbb{R}^2$ space. However, we do not observe the whole space, as the points $(\varphi, b) = (f(x^{(i,t)}), t)$ correspond to search points $x^{(i,t)}$ obtained during r runs of an algorithm A. We denote the set of all empirically observed points as $E \subset \mathbb{R}^2$.

A point $e_1 = (\varphi_1, b_1)$ is said to *strictly dominate* another point $e_2 = (\varphi_2, b_2)$ in the Pareto sense, denoted as $e_1 \prec e_2$, if the following condition is satisfied:

$$e_1 \prec e_2 \leftrightarrow (\varphi_1 \geq \varphi_2 \wedge b_1 < b_2) \vee (\varphi_1 > \varphi_2 \wedge b_1 \leq b_2).$$

A non-dominated subset of a set $Z \subseteq \mathbb{R}^d$ is defined as follows:

$$N(Z) = \{e \in Z \mid \nexists e' \in Z : e' \prec e\},$$

We define $L_0 = N(E)$ as the non-dominated subset of the empirically observed (target, budget) pairs. All the points from $\mathbb{R}^2$, which are visible on the plot and dominate L_0, have zero intensity, i.e. are transparent. This corresponds to $p = 0$ empirical probability of obtaining such target and budget pairs.

Next we construct the *Pareto layers* as $L_i = N(E \setminus L_{i-1})$, while there are some points left in E.[3] Let us denote the resulting number of layers as $|L|$. Note that $|L|$ is usually equal to the number of runs $|r|$, unless there are some completely coinciding runs. The empirical estimation of the success probability for each point that dominates any point from L_i, but is dominated by at least one point from L_{i-1}, should be plotted as $i/|L|$. In our implementation, the value of $i/|L|$ corresponds to the color intensity.

Each Pareto layer may be seen as a curve of a fixed success probability. Therefore, we may observe some approximation of the fixed-probability curves on the intensity plot. The examples are given in Section 3.

3 EXAMPLES OF INTENSITY PLOTS

In this section we illustrate the intensity plots which are already implemented in the IOHprofiler. These plots may be seen as the first step towards visualizing fixed probabilities.

In Fig. 3 an example plot is given, which illustrates the performance of the Random Local Search (RLS) algorithm on the LeadingOnes problem of size $n = 100$ from the fixed-target perspective. 11 runs of RLS were performed. RLS is the algorithm which keeps always the most recently found best-so-far solution. In each iteration, RLS creates one random neighbor by flipping one randomly chosen

[3] This process is similar, but not identical, to non-dominated sorting [3], as some layers may share points.

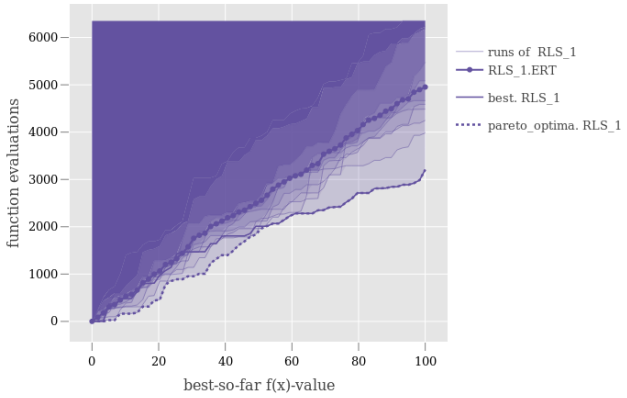

Figure 3: Intensity plot example for the Random Local Search on the LeadingOnes problem of length 100, fixed-target perspective.

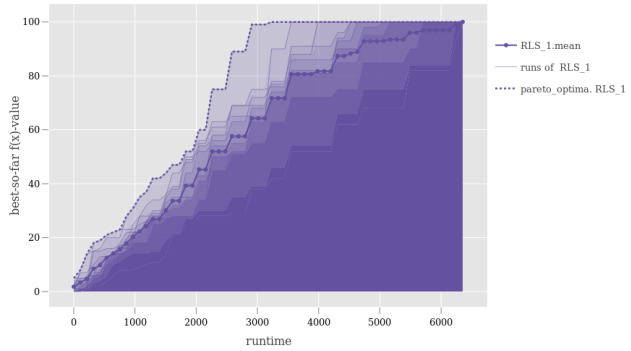

Figure 4: Intensity plot example for the Random Local Search on the LeadingOnes problem of length 100, fixed-budget perspective.

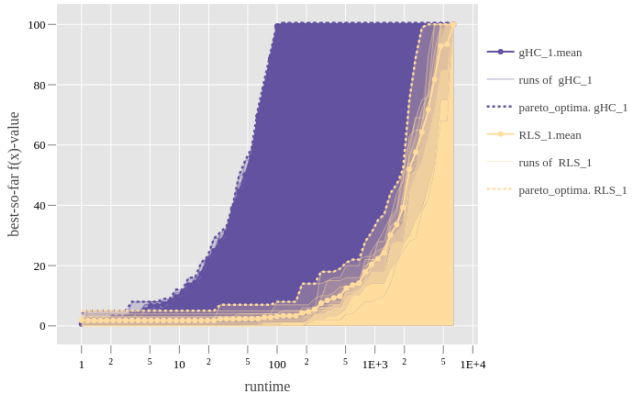

Figure 5: Comparing RLS and gHC on LeadingOnes using intensity plots.

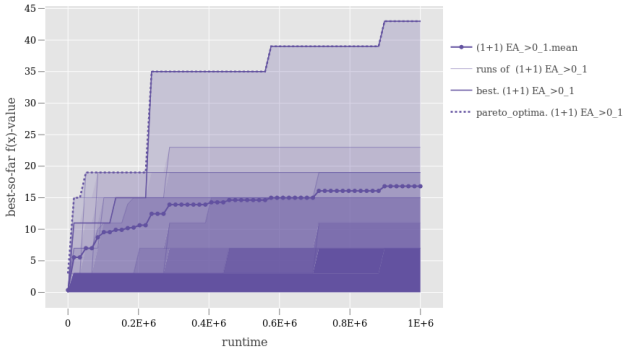

Figure 6: Intensity plot example for the $(1+1)$ evolutionary algorithm on a LeadingOnes-based W-model function.

bit. The new solution replaces the old if it is at least as good in terms of objective value. Thus, RLS is a random first-ascent hill-climber. In Fig. 3 we show the expected running time, the intensity plot, and we also explicitly show each individual run.

The darkest color depicts the area which is supposed to be reached with the 100% probability. This means that for each point (φ, b) of this area, the corresponding function value φ was always reached after b or lower number of evaluations. The area which lies below any of the observed runs is transparent, as such combinations of the function value and the number of evaluations dominate all the actually observed points. In between, we have colors of different intensity, which corresponds to the proportion of runs which reached the φ value withing the b number of evaluations or earlier. Thus we see the Pareto layers of different intensity described in Section 2.2. In future work we plan to allow the user to choose a probability value, and to show within the fixed-probability plots the respective curve that corresponds to this measure.

We also show in Fig. 3 the best run, i.e. the run with the best hitting time, and the Pareto layer L_0 defined in Section 2.2, which is denoted in the legend as "Pareto optima". It may be observed that the best run mostly lies in the light area, which means that only a little proportion of runs was characterized with such low number of function evaluations for the most of the observed function values. However, it may be noticed that the behavior of the best run was not always optimal. During roughly the first half of the optimization time, it was dominated by some other runs, thus the best run does not coincide with L_0 there.

In Fig. 4 the same data is shown from the fixed-budget perspective. The intensity of a point (b, φ) corresponds to the fraction of runs in which at least as high as φ function value was obtained after b evaluations. The "mean" notion corresponds to the average first-hitting time.

In Fig. 5 we compare RLS with a $(1+1)$ greedy hill climber (gHC). Unlike RLS, gHC does not flip a random bit in each iteration, but goes through the string from left to right and flips one bit in each iteration. After evaluating flipping the n-th bit, it starts again with flipping the bit in position one, etc. It behaves like RLS otherwise, i.e., it creates one offspring per each iteration, and keeps the most recently evaluated solution of the largest fitness. It may be clearly seen that gHC gives much better guarantees on the

obtained function value for a given budget, as it is characterized with a much bigger dark area on the intensity plot. What is more, the performance of gHC is very stable. The most of the gHC runs are close to the Pareto layer L_0, so we do not see much of a light area on its intensity plot, in contrast to the RLS plot.

Fig. 6 gives a fixed-budget intensity plot example for a different algorithm (namely, (1 + 1) EA) on a function which is harder to optimize. This function is based on a W-model transformation of LeadingOnes [10]. Compared to RLS plots, we have a limited dark area on this intensity plot, which means that a high function value was obtained by a small fraction of runs only. Note that in this example only one run reached the optimum. This run is the best one and it coincides with the Pareto layer L_0.

4 CONCLUSION

We have discussed in this work that the performance assessment of search heuristics is essentially a three-objective optimization problem, with time budget, solution quality (target value), and success probability as criteria. While fixed-budget and fixed-target performance measures are standard in today's evolutionary computation literature, fixed-probability measures are much less commonly studied. We have discussed the approximation of fixed-probability measures based on empirical results obtained from a number of runs.

We also demonstrated a first implementation towards a fixed-probability module within the framework of IOHprofiler. It is called *intensity plot* and displays the Pareto-optimal layers of (target, budget) pairs observed during an experiment. The layers are displayed using different color intensity. Our next step is to extend this module to allow the user to fix some particular success probability and to explicitly compute the corresponding fixed-probability curve.

Apart from being useful in applications where a high confidence in the success of an algorithm needs to be guaranteed, we believe that the fixed-probability measures will give additional insights into the performance of search heuristics. We note that also in the theoretical running time analysis the success probability is a measure that has recently drawn some attention, for example in terms of the p-Monte Carlo runtime notion defined in [4]. The p-Monte Carlo runtime of an algorithm A is the running time needed to find an optimal solution with probability at least $1 - p$. This corresponds to $T(A, f, \varphi = \text{opt}, 1 - p)$ in our notation from Definition 2.1. Also [9] studies fixed-probability time to optimum. Thus, overall, we are confident that fixed-probability results form a promising research topic both in empirically-oriented and in theory-oriented research streams, with the potential to complement our current view on heuristic search by a more cautious consideration of the underlying success probability.

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, in a joint call with Gaspard Monge Program for optimization, operations research and their interactions with data sciences. We also acknowledge support from COST Action CA15140 'Improving Applicability of Nature-Inspired Optimisation by Joining Theory and Practice (ImAppNIO)' supported by the European Cooperation in Science and Technology.

REFERENCES

[1] Juergen Branke, Kalyan Deb, Kaisa Miettinen, and Roman Słowiński. 2008. *Multiobjective optimization. Interactive and evolutionary approaches.* Vol. 5252.
[2] Carlos A. Coello Coello, Gary B. Lamont, and David A. Van Veldhuizen. 2006. *Evolutionary Algorithms for Solving Multi-Objective Problems* (second ed.). Springer-Verlag.
[3] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. 2002. A Fast and Elitist Multi-Objective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6, 2 (2002), 182–197.
[4] Carola Doerr and Johannes Lengler. 2017. OneMax in Black-Box Models with Several Restrictions. *Algorithmica* 78, 2 (2017), 610–640. https://doi.org/10.1007/s00453-016-0168-1
[5] Carola Doerr, Hao Wang, Furong Ye, Sander van Rijn, and Thomas Bäck. 2018. IOHprofiler: A Benchmarking and Profiling Tool for Iterative Optimization Heuristics. *arXiv e-prints:1810.05281* (Oct. 2018). arXiv:1810.05281 https://arxiv.org/abs/1810.05281 IOHprofiler is available at https://github.com/IOHprofiler.
[6] Nikolaus Hansen, Anne Auger, Dimo Brockhoff, Dejan Tusar, and Tea Tusar. 2016. COCO: Performance Assessment. *CoRR* abs/1605.03560 (2016). http://arxiv.org/abs/1605.03560
[7] N. Hansen, A. Auger, O. Mersmann, T. Tušar, and D. Brockhoff. 2016. COCO: A Platform for Comparing Continuous Optimizers in a Black-Box Setting. *ArXiv e-prints* arXiv:1603.08785 (2016).
[8] Joshua D. Knowles. 2005. A summary-attainment-surface plotting method for visualizing the performance of stochastic multiobjective optimizers. In *Proceedings of the Fifth International Conference on Intelligent Systems Design and Applications (ISDA 2005), 8-10 September 2005, Wroclaw, Poland.* IEEE Computer Society, 552–557. https://doi.org/10.1109/ISDA.2005.15
[9] Per Kristian Lehre and Dirk Sudholt. 2019. Parallel Black-Box Complexity with Tail Bounds. *CoRR* abs/1902.00107 (2019). arXiv:1902.00107 http://arxiv.org/abs/1902.00107
[10] Thomas Weise and Zijun Wu. 2018. Difficult Features of Combinatorial Optimization Problems and the Tunable W-Model Benchmark Problem for Simulating them. In *Proc. of Genetic and Evolutionary Computation Conference (GECCO'18), Companion Material).* ACM, 1769–1776. https://doi.org/10.1145/3205651.3208240
[11] Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M. Fonseca, and Viviane Grunert da Fonseca. 2003. Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Trans. Evolutionary Computation* 7, 2 (2003), 117–132. https://doi.org/10.1109/TEVC.2003.810758