Proposal for a M2 Research Internship 2018: 
Performance Criteria in Discrete Black-Box Optimization

November 29, 2017

**Supervisors:** The student will be supervised by
- Carola Doerr (CNRS and LIP6, Université Pierre et Marie Curie) and
- Benjamin Doerr (LIX, École Polytechnique)

**Contact:**
E-mail: Carola.Doerr@mpi-inf.mpg.de
Benjamin: [https://people.mpi-inf.mpg.de/~doerr/](https://people.mpi-inf.mpg.de/~doerr/)  
E-mail: doerr@lix.polytechnique.fr

**Keywords:** Randomized Algorithms, Black-Box Optimization, Search Heuristics

**General Information:** The student will have very close interactions with the two supervisors, on a day-to-day basis. The location of the internship will be the Jussieu campus of Pierre and Marie Curie University and/or the campus of École Polytechnique.

As with previous students, we aim to achieve results that are publishable in high-quality international conferences and journals.

We particularly welcome applications from students who are interested to continue their stage with a PhD under our supervision.

**Prerequisites:** The student should have a solid background in mathematics as well as some basic knowledge about algorithms and probability theory. Randomized algorithms courses would be a plus. The working language will be English, so fluency in written and spoken English is required.

**Description of the Topic:** Black-box optimization algorithms such as evolutionary algorithms, simulated annealing, and randomized local search algorithms are general-purpose optimization techniques. What distinguished them from the traditionally studied white-box counterparts is that black-box optimization algorithms do not have (or do not exploit) access to the problem instance other than by suggesting potential solution candidates and receiving (from an oracle/the black-box) information about the quality of these search points such as, for example, their function values. Based on this information, the RSH update the policy from which the next search points are sampled. This process is repeated until some stopping criterion is met. See Figure 1 for an illustration.

Despite tremendous progress in our optimization techniques in the last century, this black-box approach is still among the most frequently found approaches for the optimization of large-scale, complex, or dynamically changing real-world optimization tasks. It also plays a significant role in well-studied academic problems like numerical integration, satisfiability, scheduling, and network optimization problems, to name but a few examples. In fact, the most competitive
approaches to these problems use, to some important extend, the above-sketched black-box approach. In light of the significant efforts spent on research in optimization, this may seem unsatisfactory, as one would hope that we have put together a solid understanding of how to solve these important problems analytically. But, in practice, many—if not most—real-world optimization problems are, to date, far too complex to admit a thorough problem-specific analytical approach. In contrast, black-box optimization algorithms are typically of a rather simple structure, thus cheap to implement, and they seem to give satisfactory results for industrial needs in a short amount of time. It is therefore not surprising that the black-box approach is so frequently applied in practice.

Our goal is to build a mathematical foundation for randomized black-box optimization techniques. In this internship, we will address the question how to best evaluate the performance of black-box heuristics.

The most widely regarded performance measure for black-box optimizers is the number of oracle/black-box queries that an algorithm needs until it evaluates for the first time an optimal solution candidate. This number is called the runtime of the algorithm, or its optimization time. Optimization time is a single number that is easy to measure and that allows a quick comparison of different techniques. However, we easily observe two drawbacks:

- First, it can be used only for functions for which an optimal solution or its function value is known. While this is often the case for artificially designed problems, for many real-world problems we do not have this information.

- A second disadvantage of running time is that it does not give any information about the anytime behavior of the algorithm; e.g., how the quality of a best-so-far solution evolves over time. This information, in turn, can be quite crucial for the design of efficient parameter control techniques. To illustrate this, we plot in Figure 2 for a classic benchmark problem called LeadingOnes how many function evaluations (y-axis) an algorithm instantiated with three different parameter values \( k = 1, 2, 3 \) needs, on average, to reach solutions of different function values (x-axis). The parameter value \( k = 1 \) achieving the best overall optimization time is not the one that performs better at every step. In fact, its performance is worse for all but the last 20% of the quality targets. Picking in each step the best of the three parameter values, we obtain the lowermost, black line, which has the best anytime performance and a total optimization time that is 19% smaller than what the best static parameter choice \( k = 1 \) offers.

The situation illustrated in Figure 2 is not artificial, but can be observed for many optimization problems. It explains why we need to take a much more detailed look at performance if we want to design powerful heuristic methods.

The goal of this internship is to extend existing runtime results for black-box optimization heuristics to statements about their anytime behavior. To this end, we will study two complementary performance indicators, one measuring the quality that one can expect to obtain within a certain time budget (fixed-budget perspective), the other counting the expected number of function evaluations needed to hit intermediate function values (the plot in Figure 2 uses this fixed-target perspective). We will also discuss how to combine these two measures into runtime profiles.

Figure 2: 19% reduction of optimization time through parameter control