Algorithmica (2019) 81:589–592
https://doi.org/10.1007/s00453-018-00543-8

Preface to the Special Issue on Theory of Genetic and Evolutionary Computation

Carola Doerr[1] · Dirk Sudholt[2]

Published online: 16 January 2019

Evolutionary algorithms (EAs) are general-purpose optimizers that mimick principles from the natural evolution of species. They maintain a collection of possible solutions (the population) and then apply operators like mutation and/or recombination to create new solutions (the offspring). A selection process then chooses a new population for the next generation. Evolutionary algorithms are popular in practice as they can be easily applied to various problems. In contrast to problem-specific algorithms they require minimal or no knowledge about the problem in hand, as long as there is a way to quantify the solution quality (fitness) of new solutions. A challenge when applying EAs is that their performance and their dynamic behavior is not well understood. To remedy this, theoretical computer scientists employ methods from the analysis of randomized algorithms to analyze the performance of EAs with mathematical rigor. The aim is to develop a fundamental understanding of these algorithms and to aid in the design of new and more effective EAs. The theory track of the annual ACM *Genetic and Evolutionary Computation Conference (GECCO)* is the first tier event for advances in this direction.

In this special issue nine selected papers from the 2017 edition of the GECCO theory track are collected, each one of them carefully revised and extended to meet the high quality standards of *Algorithmica*.

Many evolutionary algorithms use an *offspring population*; i.e., in each iteration they evaluate the quality of several solution candidates, so as to profit from paralleling their quality assessment. The quality of these search points determines where and how the search is continued. One of the simplest algorithms using such a population-based approach is the (1+λ) Evolutionary Algorithm, shortly (1+λ) EA. It is known that EAs with large offspring populations can benefit from an increased mutation rate as λ offspring can amplify the chances of making good progress in one generation. But

✉ Carola Doerr
carola.doerr@mpi-inf.mpg.de

Dirk Sudholt
d.sudholt@sheffield.ac.uk

1 CNRS, Laboratoire d'informatique de Paris 6 (LIP6), Sorbonne Université, Paris, France

2 Department of Computer Science, University of Sheffield, Sheffield, UK

analytically determining the optimal mutation rate is a challenging task, in particular since it is not necessarily stable throughout the whole optimization process. In *The* $(1 + \lambda)$ *Evolutionary Algorithm with Self-Adjusting Mutation Rate* Doerr, Gießen, Witt and Yang propose and analyze a simple mechanism able to discover and track good mutation rates automatically during the run of the algorithm. The algorithm generates half the offspring with a lower mutation rate and the other half with a higher mutation rate than the current one. The current mutation rate is then adjusted with a preference towards the rate that brought forward the best offspring. For the classic OneMax benchmark problem the authors prove that this simple mechanism yields asymptotically best possible average performance amongst all λ-parallel unary unbiased black-box algorithms, that is, all search algorithms using unbiased mutation operators that create λ search points in parallel.

Classic evolutionary algorithms work with an explicit representation of the current best solution candidates, and create new solutions by mutation or recombining previously evaluated ones. Estimation of distribution algorithms (EDAs), in contrast, maintain one or more probability distributions over the entire search space. New solution candidates are sampled from these distributions and their quality determines how the distributions are updated for the next iteration. The Univariate Marginal Distribution Algorithm (UMDA) is an EDA that performs component-wise updates. Despite being one of the simplest EDA variants, its mathematical analysis turns out to be challenging already for well-structured optimization benchmarks. The two papers *Upper Bounds on the Running Time of the Univariate Marginal Distribution Algorithm on OneMax* by Witt and *Level-based Runtime Analysis of the Univariate Marginal Distribution Algorithm* by Dang, Lehre, and Nguyen both present new parameter-dependent upper bounds on the performance of the UMDA on the OneMax function. The two papers differ not only in the parameter ranges for which the results are derived, but more crucially also in the techniques that are used to prove the runtime statements. Witt provides a detailed analysis of the intricate dynamics on each bit. In particular, he identifies two distinct parameter regimes leading to expected optimization times $O(n \log n)$ on OneMax and a phase transition in the behavior of the algorithm. This leads to novel insights into the dynamic behavior of the UMDA. Dang et al. apply the so-called level-based theorem, a general analysis technique for population-based stochastic processes, along with anti-concentration bounds. Upper bounds are obtained for three different functions: OneMax, LeadingOnes, and BinVal.

An important application area of evolutionary algorithms are problems exposing some sort of uncertainty, e.g., unknown problem representation, noisy function evaluations, and dynamically changing objective functions. Four papers in this special issue address questions stemming from such uncertain problem environments.

The contribution *Solving Problems with Unknown Solution Length at Almost No Extra Cost* considers the optimization of two classic benchmark problems OneMax and LeadingOnes with the (1+1) EA. While the performance of this algorithm is quite well understood on these two functions, it was not known how well the algorithm performs when the problems are embedded in larger strings with possibly many positions that do not influence the quality of the solution candidates. Doerr, Doerr, and Kötzing prove that the resulting overhead is surprisingly small, even if no information about the position of the relevant bits is known (or explicitly learned) by the algorithm.

Uncertainty in the form of noisy function evaluations is studied by Qian, Bian, Jiang, and Tang in their work *Running Time Analysis of the (1+1)-EA for OneMax and LeadingOnes under Bit-wise Noise*. Extending the previous 1-bit noise model, in which one randomly chosen bit of the solution candidate is flipped before the quality evaluation, the authors regard a noise model in which every bit is flipped independently with some positive probability p. It is shown that the (1+1) EA can be efficient for small noise levels, but fails to be efficient when the noise level increases. The threshold between efficient and non-efficient performance can be shifted significantly when solution candidates are evaluated several times.

Another contribution studying noisy function evaluations is offered by Gavenčiak, Geissmann, and Lengler in *Sorting by Swaps with Noisy Comparisons*. Unlike most existing works in the theory of randomized search heuristics literature, which often focus on pseudo-Boolean optimization problems, the authors consider a permutation-based problem. Precisely, they study how noisy evaluations affect the classical sorting problem when tackled by an algorithm that in each iteration compares two randomly chosen elements and swaps them if they are in the wrong order. However, the noisy comparison returns an incorrect result with a fixed probability p. The authors provide a theoretical analysis for two cases: if the elements considered are always neighbored to each other, the algorithm approaches a stationary distribution of high quality in expected time $\Theta(n^2)$. If elements can have arbitrary distances, the algorithm mixes faster, but the stationary distribution is much worse, in terms of several performance indicators. Experiments complete the picture by studying intermediate models where the elements can have arbitrary distances up to a parameter r.

One of the most important challenges in the optimization of real-world problems is to handle the (often numerous) constraints which restrict the space of admissible solutions. This topic is gaining increasing interest in the evolutionary computation literature. The mathematical analysis of constrained black-box optimization problems, however, is largely unexplored. The work *Reoptimization Time Analysis of Evolutionary Algorithms on Linear Functions Under Dynamic Uniform Constraints* offers a mathematical study of optimization problems with dynamic constraints. Shi, Schirneck, Friedrich, Kötzing, and Neumann show that several classical evolutionary algorithms are quite efficient in locating new optimal solutions when the uniform constraint of an otherwise linear optimization problem changes. After acceptance and online publication of the manuscript the authors have found two mistakes, which will be corrected in an upcoming erratum. These mistakes concern Theorems 9 and 18, where upper bounds for the multi-objective EA and the multi-objective variant of the $(\mu + (\lambda, \lambda))$ GA of order $nD \log D$ and nD, respectively, are claimed. In the erratum both bounds will be corrected to $O(nD^2)$.

The Metropolis algorithm (MA), equivalent to the well-known simulated annealing algorithm with constant temperature, rejects worsening moves with a probability that depends on the fitness decrease and always accepts improving moves. Population genetics has studied a similar biological process called *Strong Selection Weak Mutation (SSWM)*, for which rigorous runtime results emerged recently. SSWM also rejects worsening moves similar to the MA, but it may also reject improving moves with a probability that decreases with the fitness improvement. In the paper *On the Analysis of Trajectory-Based Search Algorithms: When is it Beneficial to Reject Improvements?*

Nallaperuma, Oliveto, Pérez Heredia, and Sudholt ask when rejecting improvements may benefit search. The authors set out to construct example problems where both MA and SSWM perform differently. This task is complicated by the fact that both algorithms have the same stationary distribution. Consequently, the authors construct a family of example functions with very large mixing times such that SSWM is efficient while MA shows poor performance over a very long time, for all possible choices of the temperature parameter. The reason is that rejecting small improvements allows SSWM to follow the steepest gradient, whereas MA follows the first gradient it discovers. This enables SSWM to find a set of global optima, whereas MA tends to get stuck in local optima. Theoretical and experimental results further explore relations to best-improvement local search and first-improvement local search.

Island models are popular ways of running evolutionary algorithms on parallel hardware. Several islands evolve populations independently, possibly running on different machines. Every τ generations these islands exchange solutions along a given communication topology connecting the islands in a process called *migration* in order to coordinate their searches. The paper *Island Models Meet Rumor Spreading* by Doerr, Fischbeck, Frahnow, Friedrich, Kötzing, and Schirneck introduces a novel way of implementing communication: instead of each island broadcasting migrants to all neighboring islands, the authors use *rumor spreading*, where each island only communicates with one randomly chosen island. Randomized rumor spreading is a well-known epidemic paradigm that was shown to lead to fast, efficient, and robust dissemination in various applications. The paper shows that randomized rumor spreading is able to reduce the overall costs of the expected optimization time plus the expected amount of communication across the island model, compared to common topologies. The paper further refines previous analyses for a range of topologies and introduces results for novel topologies like binary trees.

We hope that with this special issue we further increase the interest of the general algorithms research community into evolutionary computation methods. We thank all authors for their submissions, our reviewers for their careful and detailed comments, and the *Algorithmica* team as well as the editor-in-chief Ming-Yang Kao for their excellent support.

Carola Doerr and Dirk Sudholt
Paris and Sheffield, December 2018