Automated configuration of genetic algorithms by tuning for anytime performance: hot-off-the-press track at GECCCO 2022

Ye, F.; Doerr, C.; Wang, H.; Bäck, T.H.W.; Fieldsend, J.E.

Automated Configuration of Genetic Algorithms by Tuning for Anytime Performance

Hot-off-the-Press Track at GECCCO 2022

Furong Ye
LIACS, Leiden University
Leiden, The Netherlands
f.ye@liacs.leidenuniv.nl

Carola Doerr
Sorbonne Université, CNRS, LIP6
Paris, France
Carola.Doerr@lip6.fr

Hao Wang
LIACS, Leiden University
Leiden, The Netherlands
h.wang@liacs.leidenuniv.nl

Thomas Bäck
LIACS, Leiden University
Leiden, The Netherlands
t.h.w.baeck@liacs.leidenuniv.nl

ABSTRACT

This paper summarizes our work "Automated Configuration of Genetic Algorithms by Tuning for Anytime Performance", to appear in *IEEE Transactions on Evolutionary Computation.*

Finding the best configuration of algorithms' hyperparameters for a given optimization problem is an important task in evolutionary computation. We compare in our work the results of four different hyperparameter optimization approaches for a family of genetic algorithms on 25 diverse pseudo-Boolean optimization problems. More precisely, we compare previously obtained results from a grid search with those obtained from three automated configuration techniques: iterated racing, mixed-integer parallel efficient global optimization, and mixed-integer evolutionary strategies. Using two different cost metrics, expected running time and the area under the empirical cumulative distribution function curve, we find that in several cases the best configurations with respect to expected running time are obtained when using the area under the empirical cumulative distribution function curve as the cost metric during the configuration process.

Our results suggest that even when interested in expected running time performance, it might be preferable to use anytime performance measures for the configuration task. We also observe that tuning for expected running time is much more sensitive with respect to the budget that is allocated to the target algorithms.

CCS CONCEPTS

• **Theory of computation → Random search heuristics**; **Design and analysis of algorithms**; *Bio-inspired optimization.*

KEYWORDS

evolutionary computation, automated algorithm configuration, hyperparameter optimization, performance metrics

GECCO '22 Companion, July 9–13, 2022, Boston, MA, USA

ACM ISBN 978-1-4503-9268-6/22/07.
https://doi.org/10.1145/3520304.3534075

ACM Reference Format:
Furong Ye, Carola Doerr, Hao Wang, and Thomas Bäck. 2022. Automated Configuration of Genetic Algorithms by Tuning for Anytime Performance: Hot-off-the-Press Track at GECCCO 2022 . In *Genetic and Evolutionary Computation Conference Companion (GECCO '22 Companion), July 9–13, 2022, Boston, MA, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3520304.3534075

1 SUMMARY OF THE RESULTS

To reduce the bias inherent to manual parameter setting procedures, a number of *automated algorithm configuration* techniques have been developed to assist the user with data-driven suggestions. The first family of algorithm configuration techniques were standard search heuristics such as mixed-integer evolution strategies, but more specific AC tools have been developed in recent years, among them surrogate-based models (e.g., SPOT [1], SMAC [5], MIP-EGO [9]), racing-based methods (Irace [8], F-race [2]) and optimization-based methods (ParamILS [6]).

We study in [10] the effects of automated algorithm configuration on a genetic algorithm (GA) framework, applied to 25 diverse pseudo-Boolean optimization problems. We compare the results of four different configuration techniques. Our main interest is in analyzing the influence of the cost metric that is used to score different configurations on the quality of the configuration suggested by the AC methods. We consider two different cost metrics: *minimizing the expected running time (ERT)* and *maximizing the area under the empirical cumulative distribution function curve of running times (AUC).* While minimizing ERT favors the average first hitting time of a single *fixed-target*, maximizing the AUC metric aims at optimizing *anytime performance*, which is measured across a whole set of (budget, target value) pairs. **We show that in several cases tuning for AUC yields configurations that have smaller ERT values than those that were obtained when directly tuning for ERT.**

Concretely, we build on our previous work [11] in which we analyzed a configurable framework of $(\mu + \lambda)$ GAs that scales the relevance of mutation and crossover by means of the crossover probability $p_c \in [0, 1]$. The framework creates new solution candidates by applying either mutation (with probability $1 - p_c$) or crossover (with probability p_c). This way, it can separate the influence of these operators from each other. While we have studied several operator choices in [11] via plain grid search, we consider here only one type of crossover (uniform crossover) and one type of mutation (standard bit mutation), to keep the search space manageable and to better highlight our key findings.

We compare in this work the results from [11] with those obtained from three different types of automated algorithm configuration methods, one based on iterated racing (we use Irace [8]), one surrogate-assisted technique (we use the mixed-integer parallel efficient global optimization MIP-EGO [9]), and a classic heuristic optimization method (we use the mixed-integer evolutionary strategies MIES suggested in [7]). Our testbed are the 25 functions from the pseudo-Boolean optimization (PBO) test suite, suggested in [4, 11] and available in the IOHprofiler benchmarking environment [3].

Compared to the $(1+1)$ EA and to the configurations obtained by the grid search in [11], we observe that $(1+\lambda)$ mutation-only GAs perform well on OneMax, LeadingOnes, and most of their so-called W-model extensions (see Sec. 3.7 in [4]), and on Ising-Models, whereas the $(\mu+\lambda)$ GA benefits from using crossover and different mutation rates on the more complex optimization problems. Thanks to its ability to handle conditional configuration spaces, Irace is the only method that finds configurations of $(1+\lambda)$ mutation-only GAs for problems on which these are efficient. We also observe that, on some problems, the automatic configurators cannot obtain hyperparameter settings that are as good as those provided by a simple grid search. However, our key finding is that the configuration methods can find better configurations in terms of the ERT by using AUC as the cost metric, compared to directly using ERT instead. This advantage is particularly pronounced when the budget of the GAs is small compared to the ERT obtained by the best possible configuration. In this case, ERT cannot reflect the differences among the configurations, and the anytime performance provides more information to guide the configuration process.

2 INTEREST FOR THE GECCO COMMUNITY

The paper addresses the GAs and their automated configuration. Both topics are highly relevant to the GECCO community. Also, the paper presents new observations on the performance of the GA and the behavior of the AC techniques when tuning for different objectives.

First, the study presents a practical example of applying AC techniques to understand the GA's behavior. Such an example illustrates the potential interaction between benchmarking studies and AC work. Also, the obtained benchmark data provide a baseline for analyzing the sensitivity of the performance concerning different parameters and can be useful for comparisons in future work.

Moreover, the paper also addresses topics that are relevant for the applications of AC techniques: (1) Better configurations in terms of ERT are found by using AUC as the cost metric, compared to directly using ERT as the cost metric, which indicates that a bi-objective (or even multi-objective) optimization process might be able to balance the advantage of the different cost metrics. (2) The performance comparison of three AC techniques suggest potential for improvement of these configuration methods.

Therefore, we are convinced that our work will stimulate interesting discussions and inspire new research questions with through a presentation in GECCO.

3 PUBLICATION INFORMATION

The paper has been recently been accepted by *IEEE Transactions on Evolutionary Computation* [10]. This work has not been presented in any of the previous GECCOs.

Concerning reproducibility, all our data is publicly available at https://zenodo.org/record/4823492#.Yk7goS8RphE and at the public repository of IOHanalyzer, accessible via https://iohanalyzer.liacs.nl/ ('2021-mlga' of 'PBO' dataset). The source code of our experiments is available at https://github.com/FurongYe/Configuration-of-Genetic-Algorithms.

ACKNOWLEDGMENTS

Our work was supported by the Chinese scholarship council (CSC No. 201706310143), by the Paris Ile-de-France Region, by ANR-11-LABX-0056-LMH (LabEx LMH), and by COST Action CA15140.

REFERENCES

[1] Thomas Bartz-Beielstein, Christian WG Lasarczyk, and Mike Preuß. 2005. Sequential parameter optimization. In *Proc. of Congress on Evolutionary Computation (CEC'05)*. IEEE, 773–780.

[2] Mauro Birattari, Zhi Yuan, Prasanna Balaprakash, and Thomas Stützle. 2010. F-Race and iterated F-Race: An overview. In *Experimental Methods for the Analysis of Optimization Algorithms*. Springer, 311–336.

[3] 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

[4] Carola Doerr, Furong Ye, Naama Horesh, Hao Wang, Ofer M Shir, and Thomas Bäck. 2020. Benchmarking discrete optimization heuristics with IOHprofiler. *Applied Soft Computing* 88 (2020), 106027.

[5] Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. 2011. Sequential Model-Based Optimization for General Algorithm Configuration. In *Proc. of Learning and Intelligent Optimization (LION'11)*. Springer, 507–523.

[6] Frank Hutter, Holger H. Hoos, Kevin Leyton-Brown, and Thomas Stützle. 2009. ParamILS: An Automatic Algorithm Configuration Framework. *Journal of Artificial Intelligence Research* 36 (2009), 267–306.

[7] Rui Li, Michael TM Emmerich, Jeroen Eggermont, Thomas Bäck, Martin Schütz, Jouke Dijkstra, and Johan HC Reiber. 2013. Mixed integer evolution strategies for parameter optimization. *Evolutionary Computation* 21, 1 (2013), 29–64.

[8] Manuel López-Ibáñez, Jérémie Dubois-Lacoste, Leslie Pérez Cáceres, Mauro Birattari, and Thomas Stützle. 2016. The Irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives* 3 (2016), 43–58.

[9] Bas van Stein, Hao Wang, and Thomas Bäck. 2019. Automatic Configuration of Deep Neural Networks with Parallel Efficient Global Optimization. In *Proc. of International Joint Conference on Neural Networks (IJCNN'19)*. IEEE, 1–7.

[10] Furong Ye, Carola Doerr, Hao Wang, and Thomas Bäck. 2022. Automated Configuration of Genetic Algorithms by Tuning for Anytime Performance. *IEEE Transactions on Evolutionary Computation* (2022). https://doi.org/10.1109/TEVC.2022.3159087 To appear.

[11] Furong Ye, Hao Wang, Carola Doerr, and Thomas Bäck. 2020. Benchmarking a $(\mu+\lambda)$ Genetic Algorithm with Configurable Crossover Probability. In *Proc. of Parallel Problem Solving from Nature (PPSN'20)*. Springer, 699–713.