IOHanalyzer: detailed performance analyses for iterative optimization heuristics: hot-off-the-press track @ GECCO 2022

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

IOHanalyzer: Detailed Performance Analyses for Iterative Optimization Heuristics

Hot-off-the-Press Track @ GECCO 2022

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

Diederick Vermetten
LIACS, Leiden University
Leiden, The Netherlands
d.l.vermetten@liacs.leidenuniv.nl

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

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

ABSTRACT

This paper summarizes our work "IOHanalyzer: Detailed Performance Analyses for Iterative Optimization Heuristics", to appear in *ACM Transactions on Evolutionary Learning and Optimization.*

IOHanalyzer is a new user-friendly tool for analyzing, comparing, and visualizing performance data of iterative optimization heuristics (IOHs). Key advantages of IOHanalyzer over other performance analysis packages are its highly interactive graphical user interface, which allows users to specify the performance measures, the ranges, and granularity of the displayed data that are most useful for their experiments, and the possibility to analyze not only performance traces, but also the evolution of dynamic parameters of IOHs.

IOHanalyzer can directly process performance data from the main benchmarking platforms, including the COCO platform, Nevergrad, the SOS platform, and IOHexperimenter. An R programming interface is provided for users preferring to have a finer control over the implemented functionalities.

Implemented in R and C, IOHanalyzer is fully open source and available on CRAN and GitHub. Our paper has two reproducibility badges, the one for "Artifacts Available" and the one for "Artifacts Evaluated – Functional v1.1".

CCS CONCEPTS

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

ACM Reference Format:
Hao Wang, Diederick Vermetten, Furong Ye, Carola Doerr, and Thomas Bäck. 2022. IOHanalyzer: Detailed Performance Analyses for Iterative Optimization Heuristics: Hot-off-the-Press Track @ GECCO 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/10.1145/3520304.3534071

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

ACM ISBN 978-1-4503-9268-6/22/07.
https://doi.org/10.1145/3520304.3534071

1 INTRODUCTION

Optimization problems not admitting exact solution approaches affect almost all aspects of our daily lives. They appear, for example, in product design, scheduling, data analysis, and machine learning (e.g., hyper-parameter tuning). For instance, it is sometimes important to analyze the optimization procedure when training a neural network, which helps us understand the training process.

Iterative optimization heuristics (IOHs) are algorithms designed to search for high-quality solutions of such problems. IOHs are characterized by a sequential structure, which aims to evolve good solutions by iteratively sampling the decision space. The distribution from which the solution candidates are sampled is adjusted after each iteration, to reflect the new information obtained from the last evaluations.

IOHs are often randomized concerning candidate generation and selecting the information stored from one iteration to the next. Therefore, the optimization behavior of IOHs is a highly complex system with many dependencies, which makes it difficult to predict how well a particular IOH performs on a given problem. To gather a good understanding of the performance and the search behavior of realistic IOHs and applications, we are therefore often restricted to an empirical evaluation of them, from which we may extrapolate the performance accurately.

Supporting such empirical evaluations through a systematic experimental design is one of the primary goals of *algorithm benchmarking*. Algorithm benchmarking addresses the selection of problem instances that are most suitable for an accurate performance extrapolation, the experimental setup of the data generation, the choice of the performance indicators and their visualizations, the choice of the statistics used to compare two or more algorithms, etc. In practice, those various aspects of algorithm benchmarking make it laborious and demanding for researchers to handle the details of experimentation, which calls for a standard and easy-to-use software implementation of algorithm benchmarking that would drastically reduce the manual work for practitioners.

In our work [8], we present IOHanalyzer, a versatile, user-friendly, and highly interactive platform for the assessment, comparison, and visualization of IOH's performance data in an algorithm-agnostic manner. Our **key design principles** are 1) an easy-to-use software

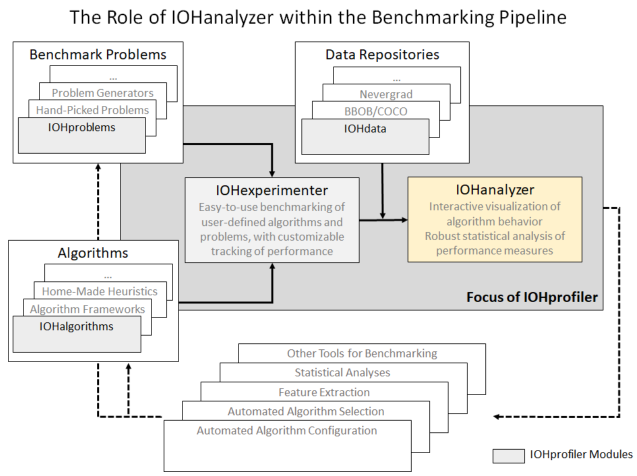

Figure 1: IOHanalyzer is a core module of the IOHprofiler benchmarking environment, which targets a complete integration of various elements of the entire benchmarking pipeline.

interface, 2) interactive performance analysis, and 3) convenient export of reports and illustrations.

IOHanalyzer is developed as the performance analysis component of the overarching IOHprofiler project - a benchmarking platform that aims to integrate various elements of the entire benchmarking pipeline, ranging from the problem (instance) generators and modular algorithm frameworks over automated algorithm configuration techniques to the actual experimentation, data analysis, and visualization [2]. An illustration of the interplay between these different components is provided in Figure 1. Notably, IOHprofiler already provides the following components:

- **IOHproblems:** a collection of benchmark problems. This component currently comprises (1) the pseudo-Boolean optimization (PBO) problems suggested in [3], (2) the integration of 24 numerical, noise-free BBOB functions from the COmparing Continuous Optimizers (COCO) platform [5], and (3) the Wmodel problem generator proposed in [9].
- **IOHalgorithms:** a collection of IOHs, e.g., randomized local search, evolutionary algorithm, fast genetic algorithm.
- **IOHdata:** a data repository for benchmark data. This repository currently comprises the data from the experiments performed in [3], a sample data set used in this paper, and some selected data sets from the COCO repository [4]. **IOHdata** also contains a data snapshot of Facebook's Nevergrad benchmarking environment [6], which is updated from the Nevergrad's data source on a regular basis.
- **IOHexperimenter:** the experimentation environment that executes IOHs on **IOHproblems** or external problems and automatically takes care of logging the experimental data. It allows for tracking the internal parameter of IOHs and supports various customizable logging options to specify when to register a data record.
- **IOHanalyzer:** the performance analysis and visualization tool presented in this work.

IOHanalyzer takes as input benchmarking data sets, generated, e.g., by IOHexperimenter, through the COCO platform, or through the Nevergrad environment. Of course, users can also use their own experimentation platform. IOHanalyzer provides an evaluation platform for these performance traces, which allows users to choose the performance measures, the ranges, and the granularity of the displayed data according to their needs. In particular, IOHanalyzer supports both a fixed-target and a fixed-budget perspective, and allows various ways of aggregating performances across different problems (or problem instances) and dimensions. In addition to these performance-oriented analyses, IOHanalyzer also offers statistics about the evolution of non-static algorithmic components, for example, the hyperparameters suggested by a self-adjusting parameter control scheme.

The R programming interface of IOHanalyzer offers a fine control on the data and functionalities implemented therein. IOHanalyzer is written in R and C and makes use of the two R packages plotly [7] and shiny [1]. The version of the software described in this paper is *v0.1.6.1*. For users less experienced with programming in R we offer a web-based graphical user interface (GUI), to which users can load their own data or use data from the **IOHdata** repository.

Availability: The stable version of the IOHanalyzer package is distributed through CRAN (https://CRAN.R-project.org/package=IOHanalyzer). The latest version is accessible on GitHub (https://github.com/IOHprofiler/IOHanalyzer, part of IOHprofiler). An up-to-date documentation is maintained on the wiki page (https://iohprofiler.github.io/). The web-based GUI of IOHanalyzer is hosted at http://iohprofiler.liacs.nl.

REFERENCES

[1] Winston Chang, Joe Cheng, JJ Allaire, Yihui Xie, and Jonathan McPherson. 2019. *shiny: Web Application Framework for R.* https://CRAN.R-project.org/package=shiny R package version 1.3.2.

[2] 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*, Article arXiv:1810.05281 (Oct 2018), arXiv:1810.05281 pages. arXiv:1810.05281 [cs.NE]

[3] Carola Doerr, Furong Ye, Naama Horesh, Hao Wang, Ofer M. Shir, and Thomas Bäck. 2020. Benchmarking discrete optimization heuristics with IOHprofiler. *Appl. Soft Comput.* 88 (2020), 106027. https://doi.org/10.1016/j.asoc.2019.106027

[4] Nikolaus Hansen, Anne Auger, and Dimo Brockhoff. 2020. Data from the BBOB workshops. https://coco.gforge.inria.fr/doku.php?id=algorithms-bbob.

[5] Nikolaus Hansen, Anne Auger, Raymond Ros, Olaf Mersmann, Tea Tušar, and Dimo Brockhoff. 2020. COCO: a platform for comparing continuous optimizers in a black-box setting. *Optimization Methods and Software* (2020), 1–31. https://doi.org/10.1080/10556788.2020.1808977 arXiv:https://doi.org/10.1080/10556788.2020.1808977

[6] Jeremy Rapin and Olivier Teytaud. 2018. Nevergrad - A gradient-free optimization platform. https://GitHub.com/FacebookResearch/Nevergrad.

[7] Carson Sievert. 2018. *plotly for R.* https://plotly-r.com

[8] Hao Wang, Diederick Vermetten, Furong Ye, Carola Doerr, and Thomas Bäck. 2022. IOHanalyzer: Detailed Performance Analyses for Iterative Optimization Heuristics. *ACM Trans. Evol. Learn. Optim.* 2, Article 3 (2022). https://doi.org/10.1145/3510426

[9] 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)* (Kyoto, Japan). ACM, 1769–1776.