The Journal of Open Source Software

Moead-framework: a modular MOEA/D Python framework

Geoffrey Pruvost[1], **Bilel Derbel**[1], **and Arnaud Liefooghe**[1]

1 Univ. Lille, CNRS, Inria, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France

DOI: 10.21105/joss.02974

Software

- Review
- Repository
- Archive

Editor: Melissa Weber Mendonça

Reviewers:

- @sjvrijn
- @chkoar

Submitted: 07 December 2020
Published: 17 October 2022

Summary

The Multi-Objective Evolutionary Algorithm based on Decomposition (MOEA/D) is a general-purpose algorithm for approximating the Pareto set of multi-objective optimization problems (Zhang & Li, 2007). It decomposes the original multi-objective problem into a number of single-objective optimization sub-problems and then uses an evolutionary process to optimize these sub-problems simultaneously and cooperatively. MOEA/D is a state-of-the-art algorithm in aggregation-based approaches for multi-objective optimization.

The goal of the *moead-framework* Python package is to provide a modular framework for scientists and researchers interested in experimenting with MOEA/D and its numerous variants.

Statement of Need

The MOEA/D algorithm is now considered as a framework. MOEA/D is the basis of many variants that improve or add new components to improve MOEA/D performance. The first version of MOEA/D and its most famous variants (Li & Zhang, 2009; Zhang et al., 2009) are implemented in recent multi-objective optimization software such as pymoo (Blank & Deb, 2020), pygmo (Biscani & Izzo, 2020) and jMetal (Nebro et al., 2015). These implementations offer many state-of-the-art algorithms, visualization tools or parallelization abstraction, but they are not modular enough to test easily all MOEA/D components. The modular R package MOEADr (Campelo et al., 2020) focuses on MOEA/D and allows the definition of different variants for each component of MOEA/D. While some modular frameworks already exist in Python for evolutionary algorithms such as DEAP (Fortin et al., 2012) or ModEA (van Rijn et al., 2016), these do not (easily) support implementing MOEA/D variants. Instead, they focus mostly on single-objective optimization and CMA-ES variants respectively.

With the *moead-framework* package, we aim to provide the modularity of the MOEADr package by using the flexibility of Python. Indeed, we want to allow the user to update the behavior of MOEA/D components in their research works without being limited by the package itself. The package is focused on a modular architecture for easily adding, updating or testing the components of MOEA/D and for customizing how components interact with each other. Indeed, in contrast with other existing implementations, *moead-framework* does not limit the users with a limited number of components available as parameters (8 components are available in MOEADr). Users can easily restructure the 10 existing components of the *moead-framework* and include new ones to easily add new features without altering existing components. Components are not only customizable with parameters as with MOEADr, but in fact they can be added with the inheritance mechanism on the main run() method of each algorithm.

For example, the *moead-framework* package was used for creating novel sub-problem selection strategies and analyzing them (Pruvost, Derbel, Liefooghe, Li, et al., 2020), and for rewriting the component used to generate new candidate (offspring) solutions with a variant based on Walsh surrogates (Pruvost, Derbel, Liefooghe, Verel, et al., 2020).

Software	Can add a new algorithm	Can modify the components of the algorithms in a modular way	Can add components to algorithms
moead-framework	yes	yes	yes
MOEADr	yes	yes	no
pymoo	yes	no	no
pygmo	yes	no	no
jMetal	yes	no	no

Documentation

The documentation is available at the following URL: moead-framework.github.io/framework/.

A complete example and all components are described in details. Two tutorials are made available for the user to experiment with their own multi-objective optimization problem and to implement their own MOEA/D variants.

Acknowledgements

This work was supported by the French national research agency (ANR-16-CE23-0013-01) and the Research Grants Council of Hong Kong (RGC Project No. A-CityU101/16).

References

Biscani, F., & Izzo, D. (2020). A parallel global multiobjective framework for optimization: pagmo. *Journal of Open Source Software*, *5*(53), 2338. https://doi.org/10.21105/joss.02338

Blank, J., & Deb, K. (2020). Pymoo: Multi-objective optimization in python. *IEEE Access*, *8*, 89497–89509. https://doi.org/10.1109/ACCESS.2020.2990567

Campelo, F., Batista, L. S., & Aranha, C. (2020). The MOEADr package: A component-based framework for multiobjective evolutionary algorithms based on decomposition. *Journal of Statistical Software*, *92*(6). https://doi.org/10.18637/jss.v092.i06

Fortin, F.-A., De Rainville, F.-M., Gardner, M.-A., Parizeau, M., & Gagné, C. (2012). DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, *13*, 2171–2175.

Li, H., & Zhang, Q. (2009). MOEA/D-DE : Multiobjective Optimization Problems With Complicated Pareto Sets, MOEA/D and NSGA-II. *IEEE Transactions on Evolutionary Computation*, *13*(2), 284–302. https://doi.org/10.1109/TEVC.2008.925798

Nebro, A. J., Durillo, J. J., & Vergne, M. (2015). Redesigning the JMetal multi-objective optimization framework. *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation*, 1093–1100. https://doi.org/10.1145/2739482.2768462

Pruvost, G., Derbel, B., Liefooghe, A., Li, K., & Zhang, Q. (2020). On the combined impact of population size and sub-problem selection in MOEA/d. In L. Paquete & C. Zarges (Eds.), *Evolutionary computation in combinatorial optimization* (pp. 131–147). Springer International Publishing. https://doi.org/10.1007/978-3-030-43680-3_9

Pruvost, G., Derbel, B., Liefooghe, A., Verel, S., & Zhang, Q. (2020). Surrogate-assisted multi-objective combinatorial optimization based on decomposition and walsh basis. *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, 542–550. https://doi.org/10.1145/3377930.3390149

van Rijn, S., Wang, H., van Leeuwen, M., & Bäck, T. (2016). Evolving the structure of evolution strategies. *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*. https://doi.org/10.1109/SSCI.2016.7850138

Zhang, Q., & Li, H. (2007). MOEA/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, *11*(6), 712–731. https://doi.org/10.1109/TEVC.2007.892759

Zhang, Q., Liu, W., & Li, H. (2009). The performance of a new version of MOEA/d on CEC09 unconstrained MOP test instances. *2009 IEEE Congress on Evolutionary Computation*, 203–208. https://doi.org/10.1109/CEC.2009.4982949