An Attempt at Predicting Algorithm Performance on Constrained Multiobjective Optimization Problems

Andrejaana Andova
Aljoša Vodopija
Jordan Cork
Tea Tušar
Bogdan Filipič
andrejaana.andova@ijs.si
Jožef Stefan Institute and Jožef Stefan International Postgraduate School
Ljubljana, Slovenia

ABSTRACT

When solving new optimization problems, it is crucial that algorithms are selected capable of both finding the best solutions and computing them in reasonable amounts of time. However, testing multiple algorithms is time-consuming and impractical. A solution to this would be to build a model that automatically selects the algorithm that performs best on a new problem. In this work, we build machine learning models to automatically predict algorithm performance on constrained multiobjective optimization problems (CMOPs) using exploratory landscape analysis (ELA) features. The results showed a high mean absolute error, which indicates that, with the currently available benchmarks and ELA features, automatically predicting algorithm performance on CMOPs is a very hard task.

KEYWORDS

constrained multiobjective optimization, evolutionary algorithms, exploratory landscape analysis, machine learning, algorithm performance prediction

1 INTRODUCTION

The common way of solving black-box constrained multiobjective optimization problems (CMOPs) is to use multiobjective optimization algorithms with constraint-handling techniques (CHTs). However, deciding which specific algorithm to use, which CHT to include, and which setting of the algorithm parameters to apply is not trivial.

In the last few years, several authors have tried to find ways of automatically selecting evolutionary algorithms for solving single-objective optimization problems [10, 13, 7]. The core concept behind their work is to extract features of benchmark single-objective optimization problems and construct a model for predicting which algorithm performs best for each individual problem. When dealing with a new problem then, the model is able to automatically decide which algorithm to use for solving the problem.

Extracting optimization problem features can be done using exploratory landscape analysis (ELA). This is a technique that takes a sample of solutions and their fitness values as input and, based on this, extracts statistical features about the problem. These features ideally characterize the problems so that similar problems have similar feature values.

In this work, we propose a first step towards automatic algorithm selection for CMOPs. This task is much harder for constrained multiobjective optimization, because, in this area, there are fewer benchmark problems available, and the ELA methods are not as well developed as in single-objective optimization.

Although the ultimate goal of our work is automatic algorithm selection, we here focus on predicting the algorithm performance of three widely used algorithms. By proposing a method for predicting algorithm performance for a few well-known algorithms, researchers can easily extend the set of algorithms, in the future.

The paper is further organized as follows. In Section 2, we introduce the theoretical background of constrained multi-objective optimization. In Section 3, we briefly describe the ELA features used in this study. In Section 4, we describe the algorithm performance measure used as the prediction target value. In Section 5, we present the experimental setup and, in Section 6, the obtained results. Finally, in Section 7, we summarize the findings and outline ideas for future work.

2 THEORETICAL BACKGROUND

A CMOP can be formulated as:

$$\begin{aligned} &\text{minimize} \quad f_m(\mathbf{x}), \quad m = 1, \ldots, M \\ &\text{subject to} \quad g_k(\mathbf{x}) \leq 0, \quad k = 1, \ldots, K, \end{aligned} \tag{1}$$

where $\mathbf{x} = (x_1, \ldots, x_D)$ is a *solution vector* of dimension D, $f_m(\mathbf{x})$ are *objective functions*, $g_k(\mathbf{x})$ are *constraint functions*, and M and K are the numbers of objectives and constraints, respectively.

In multiobjective optimization, we use the term search space S, representing a D dimensional space where all possible solution vectors $\mathbf{x}$ are located. Additionally, we can define the M dimensional objective space $P = \{f(\mathbf{x}) \mid x \in S\}$ which represents the space consisting of objective values for solutions.

A solution $\mathbf{x}$ is *feasible*, if it satisfies all constraints, $g_k(\mathbf{x}) \leq 0$, for $k = 1, \ldots, K$. A feasible solution x is said to *dominate* another feasible solution $\mathbf{y}$ if $f_m(\mathbf{x}) \leq f_m(\mathbf{y})$ for all $1 \leq m \leq M$, and $f_m(\mathbf{x}) < f_m(\mathbf{y})$ for at least one $1 \leq m \leq M$. A feasible solution $\mathbf{x}^*$ is a *Pareto-optimal solution* if there exists no feasible solution $\mathbf{x} \in S$ that dominates $\mathbf{x}^*$. All feasible solutions constitute the *feasible region* F. All nondominated feasible solutions form the *Pareto set* S_o, and the image of the Pareto set in the objective space is the *Pareto front*, $P_o = \{f(\mathbf{x}) \mid \mathbf{x} \in S_o\}$.

Nondomination ranking is a concept in multiobjective optimization that helps sort the solutions in a population into fronts, based on their dominance. Thus, all nondominated solutions get a nondomination rank of 1, solutions that are dominated only by

Information Society 2023, 9–13 October 2023, Ljubljana, Slovenia

the nondominated solutions get a nondomination rank of 2, and so on.

The point in the objective space with the best objective values is the *ideal point* $z_I = (\min_{\mathbf{x}\in S_o} f_1(\mathbf{x}), \ldots, \min_{\mathbf{x}\in S_o} f_M(\mathbf{x}))$.

The *nadir point* represents the point in the objective space with the worst fitness values across all solutions in the Pareto front $z_N = (\max_{\mathbf{x}\in S_o} f_1(\mathbf{x}), \ldots, \max_{\mathbf{x}\in S_o} f_M(\mathbf{x}))$.

The most widely used quality indicator in multiobjective optimization is the *hypervolume indicator* [17]. It maps the set of solutions found by an algorithm to a measure of the region dominated by that set and bounded by a given reference point.

3 ELA FEATURES FOR CONSTRAINED MULTIOBJECTIVE OPTIMIZATION PROBLEMS

ELA is a methodology that extracts the features of an optimization problem from a sample of its solutions. These features are usually statistical relations between the solutions and are designed by experts. Many ELA feature sets were designed for single-objective optimization problems. However, only a few feature sets exist for CMOPs.

State-of-the-art features for CMOPs were collected by Alsouly et al. [1], who adopted all of the fast-computing features for CMOPs from the related work, and also proposed some additional features. The set of all features can be divided into three groups that describe: the multiobjective landscape, the violation landscape, and the combination of the two landscapes – the multiobjective violation landscape.

All three groups of features consist of global and random walk features. The global features were calculated on a sample of size $1000 \cdot D$. The random walk features are computed during a random walk, where statistics are derived from neighboring solutions that form a sequence within the random walk. The random walk neighborhood is of size $N = 2 \cdot D + 1$, the length of the random walk is equal to $(D/N) \cdot 10^3$, and the step size is 2% of the range of the search space.

In the *multiobjective landscape group*, the features are designed to describe the objectives and the relations between them. Thus, the global features in this group include the proportion of unconstrained Pareto optimal solutions, the hypervolume of the unconstrained Pareto front, the correlation between the objective values, statistics on the unconstrained ranks, etc. The random walk features in this group include statistics on the distance between random walk neighbors in the objective space.

In the *violation landscape group*, the features are designed to describe the constraints of the problem. Thus, the global features in this group include statistics of the constraint violations, while the random walk features include statistics of the constraint violations between random walk neighbors.

In the *multiobjective violation landscape group*, the features are designed to describe the relations between the objectives and the constraints. Thus, the global features in this group include the proportion of feasible solutions, the proportion of Pareto optimal solutions, the hypervolume, statistics on the correlations between objectives and constraints, statistics on the distance between solutions in the Pareto front, etc. The random walk features in this group include statistics on the dominance relations between random walk neighbors.

Another state-of-the-art feature set for CMOPs is the one proposed by Vodopija et al. [14]. This feature set includes important information about CMOPs, including their multimodality and other landscape characteristics. However, to calculate these features one needs a larger sample size (a sample size of 250,000), which makes these features computationally very demanding.

In our study, we used both the features by Alsouly et al. and Vodopija et al.

4 EMPIRICAL CUMULATIVE DISTRIBUTION FUNCTIONS

One drawback of using hypervolume as the quality indicator in constrained multiobjective optimization is that it does not take into consideration infeasible solutions. For this reason, Vodopija et al. [15] proposed a new quality indicator designed specifically for constrained multiobjective optimization that generalizes the hypervolume-based quality indicator I_{HV+} from [5] as follows:

(1) When there are no feasible solutions in the set, the quality indicator takes on the value of the smallest constraint violation of all solutions in the set plus a threshold τ^*.
(2) When the set contains at least one feasible solution, the quality indicator equals the value of I_{HV+} bounded above by the threshold τ^*, i.e., it equals $\min\{I_{HV+}, \tau^*\}$.

The threshold value τ^* ensures that any infeasible solution will be deemed worse than any feasible one.

To measure algorithm performance during the algorithm run, we keep track of how many function evaluations, called *runtimes*, are needed to reach a particular quality indicator value, called *target*. We do so for a number of targets and visualize these runtimes using the Empirical Cumulative Distribution Function (ECDF) [5]. The ECDF measures the proportion of achieved targets at a given runtime by the given algorithm. Whenever an algorithm achieves a target, the value of the measure rises. Thus, the maximum value that can be achieved by an algorithm is equal to 1, meaning that the algorithm achieved all the targets.

In our work, we want to express algorithm performance in a single value which will serve as the target of our machine learning (ML) problem. However, the ECDF is given for any number of function evaluations (up to a maximum value). To end up with a single value, we use the area under the curve (AUC) of the ECDF, in short AUC-ECDF. This way, the ML method needs to predict a single target variable, which also includes information about the convergence of the algorithm over time. To normalize the AUC-ECDF value, we divide it by the maximum number of function evaluations.

5 EXPERIMENTAL EVALUATION

We focus on constrained bi-objective optimization problems with 2D, 3D, and 5D search spaces and, thus, use three widely used benchmarks for constrained multiobjective optimization – MW [9], CF [16], and C-DTLZ [6]. Because some benchmark problems are only defined for more than 3D or more than three objectives, the total number of problems per dimension differs. Specifically, for 2D, we have 8 out of 14 MW problems, 0 out of 10 CF problems, and 5 out of 6 C-DTLZ problems. For 3D, we have 14 out of 14 MW problems, 5 out of 10 CF problems, and 6 out of 6 C-DTLZ problems. For 5D, we have 14 out of 14 MW problems, 7 out of 10 CF problems, and 6 out of 6 C-DTLZ problems.

The focus of this work is on predicting the algorithm performance of three multiobjective optimization algorithms – NSGA-III [6], MOEA/D-IEpsilon [4], and C-TAEA [8]. Each algorithm is equipped with a different constraint-handling technique. Due to the stochastic nature of the algorithms, we conduct 31 individual runs of each algorithm on every given problem. This approach

allows us to obtain more precise values for algorithm performance. The target of the ML task for each problem is the mean AUC-ECDF value over all 31 runs of the algorithm. To facilitate the comparison of results, we use the same parameter settings for all algorithms – the population size $100 \cdot M$, and the number of generations $60 \cdot D$.

The ELA features are calculated stochastically; each time the feature calculation is started, a different sample of solutions is selected. To handle this, we created 30 samples using the Latin hypercube sampling method, resulting in 30 sets of features (learning instances) for each problem.

Predicting algorithm performance is a regression task and, therefore, we use regression ML methods – Linear Regression, Random Forest Regression (RF Regression) [2], and Epsilon-Support Vector Regression (SVR) [3]. We also included a dummy model in the comparison, which predicts the mean value of the target variable in the training data. We utilized the scikit-learn implementations [11] of these methods with default parameter settings. We tested algorithm parameter tuning as well, but there was no significant improvement of the results.

To evaluate the performance of the ML models, we use two evaluation methodologies – leave-one-sample-out and leave-one-problem-out. In the leave-one-sample-out evaluation, we use one instance as test data and the rest of the instances (including instances from other problems) as training data. We repeat this process for each instance in our dataset and take the average mean absolute error as an evaluation metric. Since all remaining instances of the problem are used during the training of the model, we expected the results from this evaluation methodology to be overly optimistic.

The leave-one-problem-out evaluation methodology is more fairly designed. In the real world, we have no information about the target problem available in the training data. Thus, in the leave-one-problem-out evaluation, we use all instances of a problem as test data and the instances from the rest of the problems as training data. This process is repeated for each problem in the dataset and the average mean absolute error is used as an evaluation metric.

6 RESULTS

The results showed a mean absolute error in the leave-one-sample-out evaluation lower than 0.01. This result is overly optimistic and shows that same-problem instances are similar to each other.

The results obtained in the leave-one-problem-out evaluation are presented in Table 1. These results show that none of the ML models performs significantly better than the dummy model. Moreover, because the target variable was normalized to [0,1], a mean absolute error between 0.09 and 0.22 is large. This indicates that the tested models trained on the current benchmarks with the current ELA features cannot be used to predict algorithm performance accurately. Also, we note that for each problem dimensionality, there is a different ML method that performs best. For 2D problems this is Linear Regression, for 3D problems RF Regression, and for 5D problems SVR.

A significantly worse performance is achieved by Linear Regression on 5D problems. When attempting to understand the cause of this, we noticed that Linear Regression achieves similar results to the other models for all problems except for one, for which it performs very poorly. A possible explanation for this could be that Linear Regression is a simple and unbounded regression method, and when a problem is different from the rest in the training set it predicts very high target values, which increase the mean absolute error.

Table 1: Mean absolute error of the predicted AUC-ECDF values with respect to the true values for 2D, 3D, and 5D problems in leave-one-problem-out evaluation.

Dim	ML method	NSGA-III	MOEA/D	C-TAEA
2D	Dummy	0.18	0.17	0.18
	Linear Regression	0.16	0.14	0.18
	RF Regression	0.19	0.18	0.18
	SVR	0.20	0.21	0.19
3D	Dummy	0.14	0.12	0.13
	Linear Regression	0.22	0.12	0.15
	RF Regression	0.12	0.09	0.11
	SVR	0.14	0.12	0.13
5D	Dummy	0.14	0.10	0.12
	Linear Regression	0.70	0.42	0.75
	RF Regression	0.13	0.09	0.12
	SVR	0.10	0.09	0.10

To better understand why the ML models performed poorly under the leave-one-problem-out evaluation, we used t-SNE [12] to reduce the dimensionality of the ELA features to 2D and visualized the results, as shown in Figure 1. Here we notice that samples from the same problem form clusters. This explains why the results of the leave-one-sample-out evaluation are significantly better than the leave-one-problem-out evaluation – in the former case, the ML task transforms into predicting the specific problem to which a sample belongs.

Analyzing the colors indicating the AUC-ECDF values of the three algorithms in Figure 1, we notice all algorithms perform similarly on almost all problems. This raises the question of whether a different algorithm parameter setting should be considered, emphasising the differences in the performance of the algorithms. For example, we could check the algorithm performance on a smaller number of generations.

When analyzing the colors showing the AUC-ECDF values of an algorithm in a single dimension, we notice there is no visible pattern. This holds for each problem dimension-algorithm combination. Notably, we often find high and low AUC-ECDF values appearing close to each other in the plot.

The results show that, with the current benchmarks and ELA features, predicting algorithm performance is very difficult.

7 CONCLUSION

In this work, we attempted to predict the algorithm performance on CMOPs, using three well-known multiobjective optimization algorithms. For this purpose, we used ELA features specially designed for CMOPs as inputs to a ML model. To calculate the ELA features, we used 30 samples for each problem, resulting in 30 learning instances per problem. The target of prediction was the algorithm's AUC-ECDF value, computed using the quality indicator designed explicitly for constrained multiobjective optimization [15].

We tested three ML regression methods – Linear Regression, RF Regression, and SVR. To compare the results from these methods, we also used a dummy model, which always predicts the mean value of the target variable in the training data. To evaluate the results, we used two evaluation methodologies – leave-one-sample-out and leave-one-problem-out.

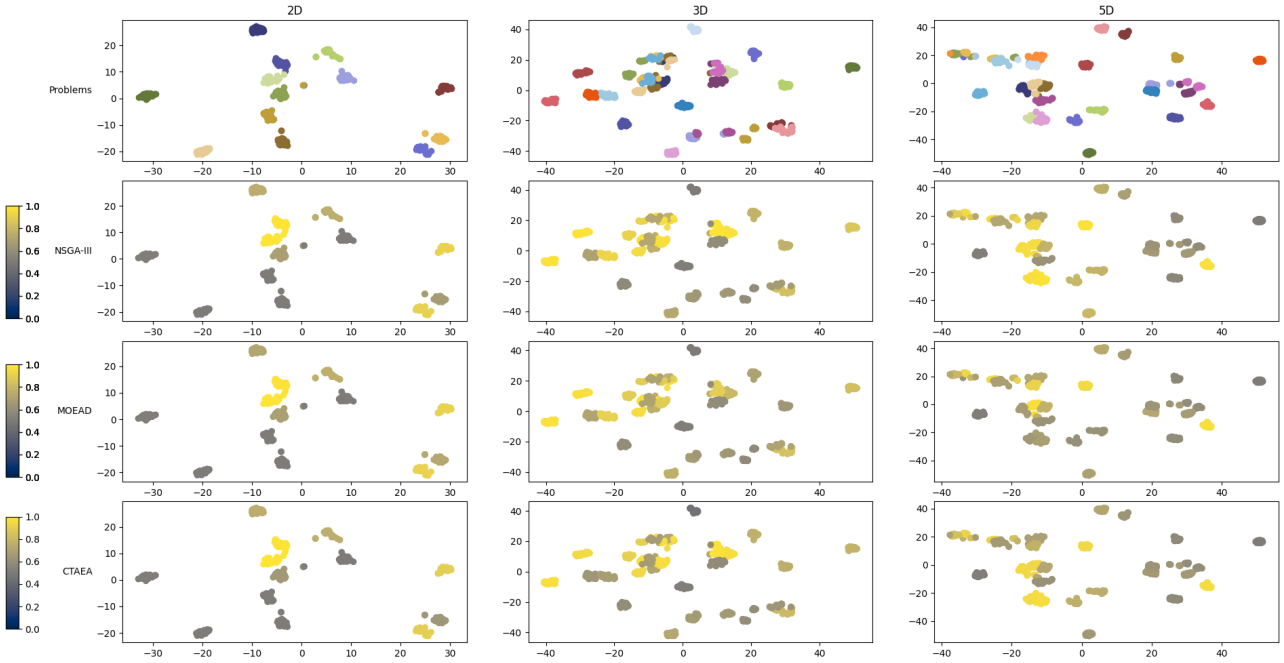

Figure 1: t-SNE visualizations of 2D, 3D and 5D problems. The colors in the first row of the plots represent the problems included in the benchmark. In the remaining rows, the colors represent the algorithm performance measured by AUC-ECDF for each algorithm considered.

In the leave-one-sample-out evaluation, very optimistic results were found, with a mean absolute error lower than 0.01. However, the results from the leave-one-problem-out evaluation were poor; none of the ML models significantly outperforms the dummy model. To explain why this occurs, we used the t-SNE method to reduce the dimensionality of the ELA features and plotted them in a color scheme indicating the performance of the algorithms. These visualizations show no visible patterns in the algorithm performance figures. Thus, we conclude that, with the currently available ELA features and benchmark problems, predicting algorithm performance is a hard task.

In future work, we aim to address two distinct aspects of the problem. The first is to improve the ELA features via automatic construction using an end-to-end deep neural network. The second is to reduce the complexity of the ML task by simplifying the target. This could be achieved by changing the task to a classification task or by changing the target to the number of generations required for an algorithm to reach a feasible solution.

ACKNOWLEDGEMENTS

The authors acknowledge the financial support from the Slovenian Research and Innovation Agency (young researcher program, research core funding No. P2-0209, and project No. N2-0254 “Constrained Multiobjective Optimization Based on Problem Landscape Analysis”).

REFERENCES

[1] Hanan Alsouly, Michael Kirley, and Mario Andrés Muñoz. 2022. An instance space analysis of constrained multi-objective optimization problems. *IEEE Transactions on Evolutionary Computation*. Early Access. https://ieeexplore.ieee.org/abstract/document/9899751.

[2] Leo Breiman. 2001. Random forests. *Machine Learning*, 45, 5–32.

[3] Harris Drucker, Christopher J Burges, Linda Kaufman, Alex Smola, and Vladimir Vapnik. 1996. Support vector regression machines. *Advances in Neural Information Processing Systems*, 9.

[4] Zhun Fan, Wenji Li, Xinye Cai, Han Huang, Yi Fang, Yugen You, Jiajie Mo, Caimin Wei, and Erik Goodman. 2019. An improved epsilon constraint-handling method in MOEA/D for CMOPs with large infeasible regions. *Soft Computing*, 23, 12491–12510.

[5] Nikolaus Hansen, Anne Auger, Dimo Brockhoff, and Tea Tušar. 2022. Anytime performance assessment in blackbox optimization benchmarking. *IEEE Transactions on Evolutionary Computation*, 26, 6, 1293–1305.

[6] Himanshu Jain and Kalyanmoy Deb. 2013. An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II: Handling constraints and extending to an adaptive approach. *IEEE Transactions on Evolutionary Computation*, 18, 4, 602–622.

[7] Anja Jankovic, Tome Eftimov, and Carola Doerr. 2021. Towards feature-based performance regression using trajectory data. In *Applications of Evolutionary Computation: 24th International Conference*. Springer, 601–617.

[8] Ke Li, Renzhi Chen, Guangtao Fu, and Xin Yao. 2018. Two-archive evolutionary algorithm for constrained multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 23, 2, 303–315.

[9] Zhongwei Ma and Yong Wang. 2019. Evolutionary constrained multiobjective optimization: Test suite construction and performance comparisons. *IEEE Transactions on Evolutionary Computation*, 23, 6, 972–986.

[10] Ana Nikolikj, Carola Doerr, and Tome Eftimov. 2023. Rf+clust for leave-one-problem-out performance prediction. In *Applications of Evolutionary Computation: 26th International Conference*. Springer, 285–301.

[11] F. Pedregosa et al. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.

[12] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9, 11.

[13] Diederick Vermetten, Hao Wang, Thomas Bäck, and Carola Doerr. 2020. Towards dynamic algorithm selection for numerical black-box optimization: Investigating bbob as a use case. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, 654–662.

[14] Aljoša Vodopija, Tea Tušar, and Bogdan Filipič. 2022. Characterization of constrained continuous multiobjective optimization problems: A feature space perspective. *Information Sciences*, 607, 244–262.

[15] Aljoša Vodopija, Tea Tušar, and Bogdan Filipič. 2023. Characterization of constrained continuous multiobjective optimization problems: A performance space perspective. *arXiv preprint arXiv:2302.02170*.

[16] Qingfu Zhang, Aimin Zhou, Shizheng Zhao, Ponnuthurai Nagaratnam Suganthan, Wudong Liu, Santosh Tiwari, et al. 2008. Multiobjective optimization test instances for the CEC 2009 special session and competition, 1–30.

[17] Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M Fonseca, and Viviane Grunert Da Fonseca. 2003. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation*, 7, 2, 117–132.