Algorithm Selection for Large-Scale Multi-objective Optimization

Mustafa Mısır[1](✉) and Xinye Cai[2]

[1] Duke Kunshan University, Kunshan, China
mustafa.misir@dukekunshan.edu.cn
[2] Dalian University of Technology, Dalian, China
xinye@dlut.edu.cn

Abstract. The present study applies Algorithm Selection to automatically specify the suitable algorithms for Large-Scale Multi-objective Optimization. Algorithm Selection has known to benefit from the strengths on multiple algorithm rather than relying one. This trait offers performance gain with limited or no contribution on the algorithm and instance side. As the target application domain, Multi-objective Optimization is a realistic way of approaching any optimization tasks. Most real-world problems are concerned with more than one objective/quality metric. This paper introduces a case study on an Algorithm Selection dataset composed of 4 Multi-objective Optimization algorithms on 63 Large-Scale Multi-objective Optimization problem benchmarks. The benchmarks involve the instances of 2 and 3 objectives with the number of variables changing between 46 and 1006, Hypervolume is the performance indicator used to quantify the solutions derived by each algorithm on every single problem instance. Since Algorithm Selection needs a suite of instance features, 4 simple features are introduced. With this setting, an existing Algorithm Selection system, i.e. ALORS, is accommodated to map these features to the candidate algorithms' performance denoted in ranks. The empirical analysis showed that this basic setting with AS is able to offer better performance than those standalone algorithms. Further analysis realized on the algorithms and instances report similarities/differences between algorithms and instances while reasoning the instances' hardness to be solved.

1 Introduction

Optimization [1] is a process concerned with exploring the best solution regarding some performance criteria. These criteria are referred to objective functions that can measure the solution quality regarding a target problem. The number of objectives determine the nature of the problem. A large group of optimization research focuses on the problems with only one objective, i.e. single-objective optimization. However, the majority of the real-world applications actually come with more than one objective. Those problems are categorized as the multi-objective optimization problems (MOPs) [2]. Further categorization is possible

B. Dorronsoro et al. (Eds.): OLA 2023, CCIS 1824, pp. 36–47, 2023.
https://doi.org/10.1007/978-3-031-34020-8_3

when the number of objectives is exactly two, i.e. bi-objective optimization. If the count exceeds three, then the MOPs are denoted as many-objective optimization. [3,4].

The main challenge of having multiple objectives is that they are likely to be conflicting. Improving one objective can degrade the quality of the remaining objectives. This leads to solution quality evaluation based on various performance indicators utilizing all the objectives. R2 [5,6], Hyper-volume (HV) [7], Generational Distance (GD) [8], Inverted/Inverse GD (IGD) [9], IGD+ [10], Spread [11], and Epsilon [12] are well-known examples of the performance indicators. These indicators are mostly linked to Pareto fronts (PFs) where multiple solutions are maintained. PFs consist of the solutions that do not strictly dominate any other solution, i.e. the solutions that are not worse than the remaining solutions considering all the objectives. In that respect, the algorithms developed for the MOPs mostly operate on the populations of solutions, i.e. the population-based algorithms. Multi-objective Evolutionary Algorithms (MOEAs) [13,14] take the lead in that domain. Non-dominated Sorting Genetic Algorithm II (NSGA-II) [11,15], Pareto Archived Evolution Strategy (PAES) [16], Strength Pareto Evolutionary Algorithm 2 (SPEA2) [17], Pareto Envelope-based Selection Algorithm II (PESA-II) [18] and MOEA based on Decomposition (MOEA/D) [19] are some examples from the literature. There are other population-based algorithms besides MOEAs, using meta-heuristics like Particle Swarm Optimization (MOPSO) [20] and Ant Colony Optimization [21]. It is also possible to see their hybridized variants [22–24].

Despite these immense algorithm development efforts, it is unlikely to see a truly best, i.e. always coming first, algorithm on the existing benchmark scenarios under fair experimental conditions. This practical fact is further supported theoretically by the No Free Lunch (NFL) theorem [25]. This study focuses on automatically determining the algorithm to be applied for each given MOP instance, through Algorithm Selection (AS) [26]. AS is a meta-algorithmic approach offering improved performance through selection. The idea is to automatically choose algorithms from given problem solving scenarios. The selection operations are carried on a given algorithm set [27] consisting of those candidate methods to be picked. The traditional way of approach AS is in the form of performance prediction models. In that respect, a suite of features is needed to characterize the target problem instances. These features are matched with the performance of the candidate algorithms on a group of training instances. While the use of human-engineered features is common for AS, Deep Learning (DL) has also been used for automatically extracting features [28].

AS has been applied to a variety of problem domains such as Boolean Satisfiability (SAT) [29] Constraint Satisfaction (CSP) [30], Blackbox Optimization [31], Nurse Rostering (NRP) [32], Graph Coloring (GCP) [33], Traveling Salesman (TSP) [34] Traveling Thief Problem (TTP) [35], and Game Playing [36]. AS library (ASlib) [37] provides a diverse and comprehensive problem sets for AS. There have been development efforts of new AS systems for addressing these problems. SATzilla [29] is a well known AS method, particularly popularized due to its success in the SAT competitions. Hydra [38] is an example aiming at constructing algorithm sets, a.k.a. Algorithm Portfolios [27], via configuring the given algorithms. The portfolio building task has been studied for different selection tasks [39–42]. 3S [43] delivers algorithm schedules, assigning runtime to the algorithms for each given problem instance. Unlike these AS level contributions, Autofolio [44] takes the search to a higher level by seeking the best AS setting of varying components and parameter configurations. As another high-level approach, AS is used for performing per-instance selection across Selection Hyper-heuristics (SHHs) [45].

The present study performs AS to identify suitable algorithms for the given MOP instance. To be specific, the problem targeted here is the Large-scale MOP (LSMOP) where the number of decision variables can reach up to the vicinity of thousands. The instance set is based on 9 LSMOP benchmarks. Those base benchmarks are varied w.r.t. the number of objectives, i.e. 2 or 3, and the number of decision variables, varies between 46 and 1006, leading to 63 LSMOP instances. The task is to perform per-instance AS using an existing AS system named ALORS [46], among 4 candidate population-based algorithms. Hypervolume (HV) is used as the performance indicator. Experimental analysis carried out illustrated that AS only with 4 basic features outperforms those constituent multi-objective algorithms.

In the remainder of the paper, Sect. 2 discusses the use of AS. An empirical analysis is reported in Sect. 3. Section 4 comes with the concluding remarks besides discussing the future research ideas.

2 Method

ALORS [46] is concerned with the selection task as a recommender system (RS). ALORS specifically uses Collaborative Filtering (CF) [47] in that respect. Unlike the existing AS systems, ALORS is able to perform with the sparse/incomplete performance data, M, while maintaining high, comparable performance to the complete data. The performance refers to running a set of algorithms, A, on a group of instances, I. Thus, the performance data is a matrix of $M_{|I| \times |A|}$. For decreasing the data generation cost of such sparse data has been further targeted in [48,49]. While the entries of the performance data vary from problem to problem, ALORS generalizes them by using the rank data, $\mathcal{M}$. Thus, any given performance data is first converted into rank data. Unlike the traditional AS systems, ALORS builds a prediction model with an intermediate feature-to-feature mapping step, instead of providing a direct rank prediction. The initial, hand-picked/designed features are referenced to a set of latent (hidden) features. These features are extracted directly from the rank performance data by using Singular Value Decomposition (SVD) [50]. SVD is a well-known Matrix Factorization (MF) strategy, used in various CF based RS applications [51]. SVD returns two matrices, U and V besides a diagonal matrix accommodating the singular values as $\mathcal{M} = U \Sigma V^t$. U represents the rows of $\mathcal{M}$, i.e. instances, while V displays its columns, i.e. algorithms, similarly to [52,53]. Beyond representing those data elements, the idea is the reduce the dimensions, $r \leq min(|I|, |A|)$, hopefully eliminating the possible noise in $\mathcal{M}$.

$$\mathcal{M} \approx U_r \varSigma_r V_r^t$$

ALORS maps a given initial set of instance features F to U_r. The predicted performance ranks are calculated by multiplying U_r with the remaining matrices of $\varSigma_r$ and V_r^t. In that respect, for a new problem instances, ALORS essentially determines an array of values, i.e. a new row for U_r. Its multiplication with $\varSigma_r$ and V_r^t delivers the expected performance ranks of the candidate algorithms on this new problem instance.

3 Computational Results

Despite the capabilities of ALORS as the sole Algorithm Selection (AS) approach, on working with incomplete performance data, the instance × algorithm rank data here has the complete performance entries. The AS data is directly derived from [54]. The data on the Large-Scale Multi-objective Optimisation Problem (LSMOP) consists of 4 algorithms. The candidate algorithms are Speed-constrained Multi-objective Particle Swarm Optimization (SMPSO) [55], Multi-objective Evolutionary Algorithm based on Decision Variable Analysis (MOEA/DVA) [56], Large-scale Many-objective Evolutionary Algorithm (LMEA) [57] and Weighted Optimization Framework SMPSO (WOF-SMPSO) [58]. The hypervolume (HV) indicator [59] is used as the performance metric.

Table 1. The base LSMOP instances

Problem	Modality	Separability
LSMOP1	Unimodal	Fully Separable
LSMOP2	Mixed	Partially Separable
LSMOP3	Multi-modal	Mixed
LSMOP4	Mixed	Mixed
LSMOP5	Unimodal	Fully Separable
LSMOP6	Mixed	Partially Separable
LSMOP7	Multi-modal	Mixed
LSMOP8	Mixed	Mixed
LSMOP9	Mixed	Fully Separable

Table 1 shows the specifications of the LSMOP benchmark functions [60]. The functions differ in terms of modality and separability. The 2-objective and 3-objective variants of each function are considered. Besides that further variations on the functions are achieved using different number of decision variables. In total, 63 LSMOP instances are present. The instances are encoded as LSMOPX_m=a_n=b where X is the base LSMOP index, m refers to the number of objectives and n is for the number of decision variables. All these instances are represented using just 4 features. Besides the modality and separability characteristics, the number of objectives and the number of variables are as the instance features.

Table 2. The average ranks of each constituent algorithm besides ALORS where the best per-benchmark performances are in bold (AVG: the average rank considering the average performance on each benchmark function; O-AVG: the overall average rank across all the instances)

Benchmark	SMPSO	MOEA/DVA	LMEA	WOF-SMPSO	ALORS
LSMOP1	4.57	**1.71**	3.71	3.07	1.93
LSMOP2	4.43	**2.43**	3.14	2.5	2.5
LSMOP3	4.86	2.43	3	**2.36**	**2.36**
LSMOP4	4.43	2.79	3.71	2.57	**1.5**
LSMOP5	3.14	**1.86**	5	2.93	2.07
LSMOP6	3.86	3.5	3.64	**1.21**	2.79
LSMOP7	3.71	3.79	4.29	**1.29**	1.93
LSMOP8	4	2.93	4.43	2.14	**1.5**
LSMOP9	2.86	**1.79**	4.43	3.86	2.07
AVG	3.98 ± 0.67	2.58 ± 0.74	3.93 ± 0.66	2.44 ± 0.84	**2.07 ± 0.43**
O-AVG	3.98 ± 1.04	2.58 ± 1.30	3.93 ± 1.19	2.44 ± 1.19	**2.07 ± 0.89**

Table 2 reports the performance of all the candidate algorithms besides ALORS as the automated selection method. Average performance on all the instances show that ALORS offers the best performance with the average rank of 2.07. The closest approach that is the single best method, i.e. WOF-SMPSO, comes with the average rank of 2.44 while SMPSO shows the overall worst performance with the average rank of 3.98. Referring to the standard deviations, ALORS also comes with the most robust behaviour.

Figure 1 reports the selection frequencies of each constituent algorithm. Oracle denotes the optimal selection, i.e. choosing the best algorithm for each instance. The graph shows that ALORS shows similar behaviour to Oracle with minor variations. MOEA/DVA and WOF-SMPSO are the most frequently selected algorithms. Besides the pure selection frequencies, ALORS does not utilize SMPSO at all while it is preferred for two instances by Oracle.

Figure 2 illustrates the importance of each single feature in terms of Gini Index, derived by Random Forest (RF). All four features happen to contribute to the selection model. Being said that *separability* comes as the most critical feature while *modality* is the least important one.

Figure 3 reports the dis/-similarities of the LSMOP benchmark function instances. Linking to the feature importance analysis in Fig. 2, there is no a

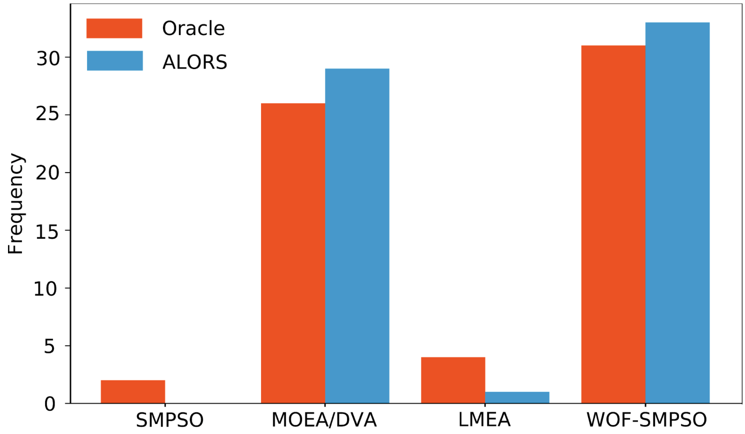

Fig. 1. The selection frequencies of each algorithm by Oracle and ALORS

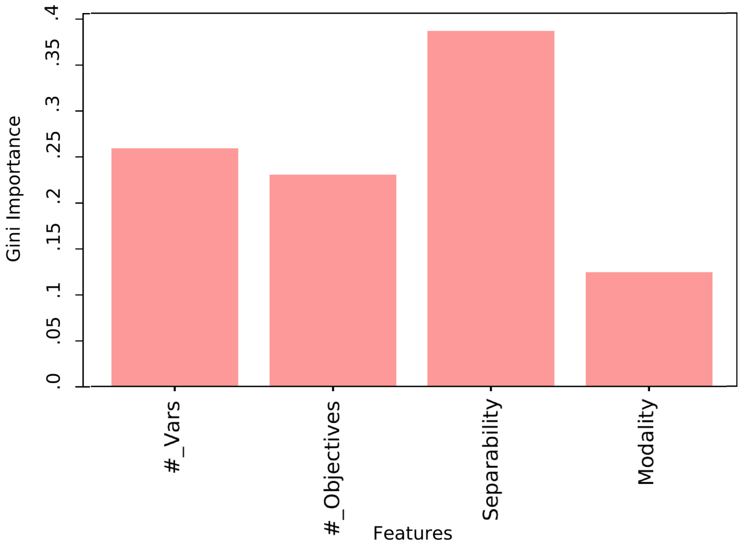

Fig. 2. Importance of the initial, hand-picked LSMOP benchmark function instance features, using Gini Index/Importance

single criterion/feature to emphasize instance dis/-similarity, yet it is still possible to see the effects of separability. As an example, consider the 10 most similar instances provided on the right bottom of the clustering figure. The instances are LSMOP1_m=2_n=46, LSMOP1_m=2_n=106, LSMOP5_m=3_n=212, LSMOP5_m=3_n=112, LSMOP5_m=3_n=52, LSMOP2_m=2_n=106, LSMOP5_m=2_n=1006, LSMOP1_m=2_n=206, LSMOP8_m=3_n=52 and LSMOP9_m=3_n=112. 8 of them are fully separable. The remaining 2 instances are partially separable and mixed, respectively. Referring to the second best fea-

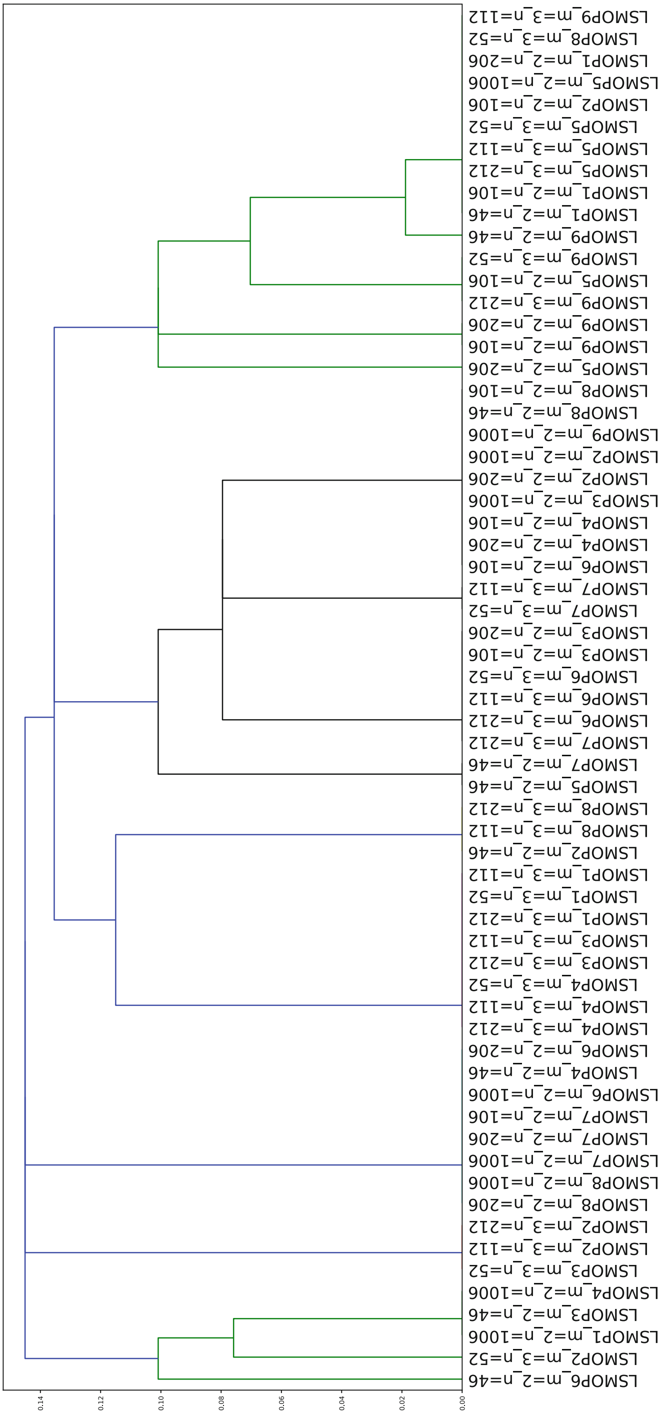

Fig. 3. Hierarchical clusters of instances using the latent features extracted from the performance data by SVD ($k = 3$)

tures, i.e. number of variables, the values change from 52 to 1006. Being said that 1006 occurs only once, thus The half of the instances have 2 objectives while the other half is with 3 objectives. As 2 out of 3 fully separable benchmark functions are unimodal, 7 instances happen to be unimodal. The other 3 instances are mixed in terms of modality.

Figure 4 illustrates the candidate algorithms which are hierarchically clustered. Referring to the best performing standalone algorithm, i.e. WFO-SMPSO, there is resemblance to SMPSO which is the base approach of WFO-SMPSO. Although their performance levels differ, their performance variations across the tested instances are similar.

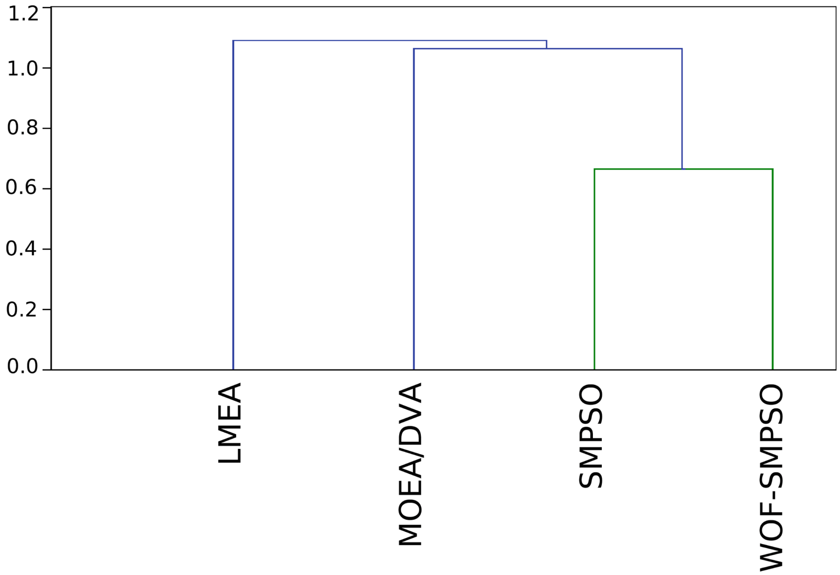

Fig. 4. Hierarchical clusters of algorithms using the latent features extracted from the performance data by SVD ($k = 3$)

4 Conclusion

This study utilizes Algorithm Selection (AS) for Large-Scale Multi-objective Optimization, using Hyper-volume (HV) as the performance criterion. Multi-objective optimization is concerned with the majority of the real-world optimization tasks. In that respect, there have been immense efforts both problem modelling and algorithm development for multi-objective optimization. However, there is no ultimate multi-objective optimization algorithm that can outperform the competing algorithms under fair experimental settings. This practical fact reveals a clear performance gap that can be filled by AS. AS suggests a way to automatically determine the best algorithms for given problem instances.

The present work performs on 4 multi-objective optimization algorithm for 63 benchmarks originated from 9 base problems. For the instance characterization required to use AS, 4 simple instance features are determined. The corresponding computational analysis showed that AS is able to suppress those candidate algorithms. Further analysis carried on the algorithm and instance spaces delivered insights on the instance hardness, instance similarity and algorithm resemblance.

As the first study of using AS for multi-objective optimization, there are a variety of research tasks to be tackled as future research. The initial follow-up work is concerned with extending both the algorithm and instance space. Additionally, the well-known multi-objective performance indicators will be incorporated. The analysis on the algorithm and instance spaces will be extended accordingly. While an AS model will be derived for each indicator, the selection will be also achieved by taking all the indicators into account like a Pareto frontier. The idea will then be reversed to devise AS a multi-objective selection problem where the performance measures are the common AS metrics such as the Par10 score and success rate.

References

1. Chong, E.K., Zak, S.H.: An Introduction to Optimization. Wiley, Hoboken (2004)
2. Deb, K., Deb, K.: Multi-objective optimization. In: Burke, E., Kendall, G. (eds.) Search Methodologies, pp. 403–449. Springer, Boston (2014). https://doi.org/10.1007/978-1-4614-6940-7_15
3. Ishibuchi, H., Tsukamoto, N., Nojima, Y.: Evolutionary many-objective optimization: a short review. In: IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), pp. 2419–2426. IEEE (2008)
4. Li, K., Wang, R., Zhang, T., Ishibuchi, H.: Evolutionary many-objective optimization: a comparative study of the state-of-the-art. IEEE Access **6**, 26194–26214 (2018)
5. Hansen, M.P., Jaszkiewicz, A.: Evaluating the quality of approximations to the non-dominated set. Citeseer (1994)
6. Brockhoff, D., Wagner, T., Trautmann, H.: On the properties of the R2 indicator. In: Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation, pp. 465–472 (2012)
7. Zitzler, E., Thiele, L.: Multiobjective optimization using evolutionary algorithms—a comparative case study. In: Eiben, A.E., Bäck, T., Schoenauer, M., Schwefel, H.P. (eds.) PPSN 1998. LNCS, pp. 292–301. Springer, Cham (1998). https://doi.org/10.1007/bfb0056872
8. Van Veldhuizen, D.A., Lamont, G.B.: Multiobjective evolutionary algorithm research: a history and analysis. Technical report, Department of Electrical and Computer Engineering Air Force Institute of Technology, OH, Technical Report TR-98-03 (1998)
9. Coello Coello, C.A., Reyes Sierra, M.: A study of the parallelization of a coevolutionary multi-objective evolutionary algorithm. In: Monroy, R., Arroyo-Figueroa, G., Sucar, L.E., Sossa, H. (eds.) MICAI 2004. LNCS (LNAI), vol. 2972, pp. 688–697. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24694-7_71

10. Ishibuchi, H., Masuda, H., Tanigaki, Y., Nojima, Y.: Modified distance calculation in generational distance and inverted generational distance. In: Gaspar-Cunha, A., Henggeler Antunes, C., Coello, C.C. (eds.) EMO 2015. LNCS, vol. 9019, pp. 110–125. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-15892-1_8
11. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**, 182–197 (2002)
12. Fonseca, C.M., Knowles, J.D., Thiele, L., Zitzler, E.: A tutorial on the performance assessment of stochastic multiobjective optimizers. In: Proceedings of the 3rd International Conference on Evolutionary Multi-Criterion Optimization (EMO), vol. 216, p. 240 (2005)
13. Deb, K.: Multi-objective optimisation using evolutionary algorithms: an introduction. In: Wang, L., Ng, A., Deb, K. (eds.) Multi-objective Evolutionary Optimisation for Product Design and Manufacturing, pp. 3–34. Springer, London (2011). https://doi.org/10.1007/978-0-85729-652-8_1
14. Deb, K.: Multi-objective evolutionary algorithms. In: Kacprzyk, J., Pedrycz, W. (eds.) Springer Handbook of Computational Intelligence, pp. 995–1015. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-43505-2_49
15. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In: Schoenauer, M., et al. (eds.) PPSN 2000. LNCS, vol. 1917, pp. 849–858. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45356-3_83
16. Knowles, J.D., Corne, D.W.: Approximating the nondominated front using the pareto archived evolution strategy. Evol. Comput. **8**, 149–172 (2000)
17. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: improving the strength pareto evolutionary algorithm. Technical Report 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland (2001)
18. Corne, D.W., Jerram, N.R., Knowles, J.D., Oates, M.J.: PESA-II: region-based selection in evolutionary multiobjective optimization. In: Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation (GECCO), pp. 283–290. Morgan Kaufmann Publishers Inc. (2001)
19. Zhang, Q., Li, H.: MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**, 712–731 (2007)
20. Coello, C.C., Lechuga, M.S.: MOPSO: a proposal for multiple objective particle swarm optimization. In: Proceedings of the IEEE Congress on Evolutionary Computation (CEC), vol. 2, pp. 1051–1056. IEEE (2002)
21. Ding, L.P., Feng, Y.X., Tan, J.R., Gao, Y.C.: A new multi-objective ant colony algorithm for solving the disassembly line balancing problem. Int. J. Adv. Manuf. Technol. **48**, 761–771 (2010)
22. Mashwani, W.K.: MOEA/D with DE and PSO: MOEA/D-DE+PSO. In: Bramer, M., Petridis, M., Nolle, L. (eds.) SGAI 2011, pp. 217–221. Springer, London (2011). https://doi.org/10.1007/978-1-4471-2318-7_16
23. Ke, L., Zhang, Q., Battiti, R.: MOEA/D-ACO: a multiobjective evolutionary algorithm using decomposition and antcolony. IEEE Trans. Cybern. **43**, 1845–1859 (2013)
24. Alhindi, A., Zhang, Q.: MOEA/D with tabu search for multiobjective permutation flow shop scheduling problems. In: Proceedings of the IEEE Congress on Evolutionary Computation (CEC), pp. 1155–1164. IEEE (2014)
25. Wolpert, D., Macready, W.: No free lunch theorems for optimization. IEEE Trans. Evol. Comput. **1**, 67–82 (1997)

26. Kerschke, P., Hoos, H.H., Neumann, F., Trautmann, H.: Automated algorithm selection: survey and perspectives. Evol. Comput. **27**, 3–45 (2019)
27. Gomes, C., Selman, B.: Algorithm portfolios. Artif. Intell. **126**, 43–62 (2001)
28. Loreggia, A., Malitsky, Y., Samulowitz, H., Saraswat, V.A.: Deep learning for algorithm portfolios. In: Proceedings of the 13th Conference on Artificial Intelligence (AAAI), pp. 1280–1286 (2016)
29. Xu, L., Hutter, F., Hoos, H., Leyton-Brown, K.: SATzilla: portfolio-based algorithm selection for SAT. J. Artif. Intell. Res. **32**, 565–606 (2008)
30. Yun, X., Epstein, S.L.: Learning algorithm portfolios for parallel execution. In: Hamadi, Y., Schoenauer, M. (eds.) LION 2012. LNCS, pp. 323–338. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34413-8_23
31. Kerschke, P., Trautmann, H.: Automated algorithm selection on continuous blackbox problems by combining exploratory landscape analysis and machine learning. Evol. Comput. **27**, 99–127 (2019)
32. Messelis, T., De Causmaecker, P., Vanden Berghe, G.: Algorithm performance prediction for nurse rostering. In: Proceedings of the 6th Multidisciplinary International Scheduling Conference: Theory and Applications (MISTA 2013), pp. 21–38 (2013)
33. Musliu, N., Schwengerer, M.: Algorithm selection for the graph coloring problem. In: Nicosia, G., Pardalos, P. (eds.) LION 2013. LNCS, vol. 7997, pp. 389–403. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-44973-4_42
34. Kotthoff, L., Kerschke, P., Hoos, H., Trautmann, H.: Improving the state of the art in inexact TSP solving using per-instance algorithm selection. In: Dhaenens, C., Jourdan, L., Marmion, M.-E. (eds.) LION 2015. LNCS, vol. 8994, pp. 202–217. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19084-6_18
35. Wagner, M., Lindauer, M., Mısır, M., Nallaperuma, S., Hutter, F.: A case study of algorithm selection for the traveling thief problem. J. Heuristics **24**, 295–320 (2018)
36. Stephenson, M., Renz, J.: Creating a hyper-agent for solving angry birds levels. In: AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (2017)
37. Bischl, B., et al.: ASlib: a benchmark library for algorithm selection. Artif. Intell. **237**, 41–58 (2017)
38. Xu, L., Hoos, H., Leyton-Brown, K.: Hydra: automatically configuring algorithms for portfolio-based selection. In: Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI), pp. 210–216 (2010)
39. Mısır, M., Handoko, S.D., Lau, H.C.: OSCAR: online selection of algorithm portfolios with case study on memetic algorithms. In: Dhaenens, C., Jourdan, L., Marmion, M.-E. (eds.) LION 2015. LNCS, vol. 8994, pp. 59–73. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19084-6_6
40. Mısır, M., Handoko, S.D., Lau, H.C.: ADVISER: a web-based algorithm portfolio deviser. In: Dhaenens, C., Jourdan, L., Marmion, M.-E. (eds.) LION 2015. LNCS, vol. 8994, pp. 23–28. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19084-6_3
41. Lau, H., Mısır, M., Xiang, L., Lingxiao, J.: ADVISER$^+$: toward a usable web-based algorithm portfolio deviser. In: Proceedings of the 12th Metaheuristics International Conference (MIC), Barcelona, Spain, pp. 592–599 (2017)
42. Gunawan, A., Lau, H.C., Mısır, M.: Designing and comparing multiple portfolios of parameter configurations for online algorithm selection. In: Festa, P., Sellmann, M., Vanschoren, J. (eds.) LION 2016. LNCS, vol. 10079, pp. 91–106. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-50349-3_7

43. Kadioglu, S., Malitsky, Y., Sabharwal, A., Samulowitz, H., Sellmann, M.: Algorithm selection and scheduling. In: Lee, J. (ed.) CP 2011. LNCS, vol. 6876, pp. 454–469. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-23786-7_35
44. Lindauer, M., Hoos, H.H., Hutter, F., Schaub, T.: AutoFolio: an automatically configured algorithm selector. J. Artif. Intell. Res. **53**, 745–778 (2015)
45. Misir, M.: Cross-domain algorithm selection: algorithm selection across selection hyper-heuristics. In: 2022 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 22–29. IEEE (2022)
46. Mısır, M., Sebag, M.: ALORS: an algorithm recommender system. Artif. Intell. **244**, 291–314 (2017)
47. Su, X., Khoshgoftaar, T.M.: A survey of collaborative filtering techniques. Adv. Artif. Intell. **2009**, 4 (2009)
48. Mısır, M.: Data sampling through collaborative filtering for algorithm selection. In: The 16th IEEE Congress on Evolutionary Computation (CEC), pp. 2494–2501. IEEE (2017)
49. Mısır, M.: Active matrix completion for algorithm selection. In: Nicosia, G., Pardalos, P., Umeton, R., Giuffrida, G., Sciacca, V. (eds.) LOD 2019. LNCS, vol. 11943, pp. 321–334. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-37599-7_27
50. Golub, G.H., Reinsch, C.: Singular value decomposition and least squares solutions. Numer. Math. **14**, 403–420 (1970)
51. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**, 30–37 (2009)
52. Mısır, M.: Matrix factorization based benchmark set analysis: a case study on HyFlex. In: Shi, Y., et al. (eds.) SEAL 2017. LNCS, vol. 10593, pp. 184–195. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68759-9_16
53. Mısır, M.: Benchmark set reduction for cheap empirical algorithmic studies. In: Proceedings of the IEEE Congress on Evolutionary Computation (CEC) (2021)
54. Zille, H., Mostaghim, S.: Comparison study of large-scale optimisation techniques on the LSMOP benchmark functions. In: IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1–8. IEEE (2017)
55. Nebro, A., Durillo, J., García-Nieto, J., Coello Coello, C., Luna, F., Alba, E.: SMPSO: a new PSO-based metaheuristic for multi-objective optimization. In: IEEE Symposium on Computational Intelligence in Multicriteria Decision-Making (MCDM 2009), pp. 66–73. IEEE Press (2009)
56. Ma, X., et al.: A multiobjective evolutionary algorithm based on decision variable analyses for multiobjective optimization problems with large-scale variables. IEEE Trans. Evol. Comput. **20**, 275–298 (2015)
57. Zhang, X., Tian, Y., Cheng, R., Jin, Y.: A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization. IEEE Trans. Evol. Comput. **22**, 97–112 (2016)
58. Zille, H., Ishibuchi, H., Mostaghim, S., Nojima, Y.: A framework for large-scale multiobjective optimization based on problem transformation. IEEE Trans. Evol. Comput. **22**, 260–275 (2017)
59. Auger, A., Bader, J., Brockhoff, D., Zitzler, E.: Hypervolume-based multiobjective optimization: theoretical foundations and practical implications. Theor. Comput. Sci. **425**, 75–103 (2012)
60. Cheng, R., Jin, Y., Olhofer, M., et al.: Test problems for large-scale multiobjective and many-objective optimization. IEEE Trans. Cybern. **47**, 4108–4121 (2016)