Multi-Objective Self Regulating Particle Swarm Optimization Algorithm for BMOBench Platform

M. R. Tanweer[1,2], A. Al-Dujaili[2], and S. Suresh[2]

[1]Hamdard Institute of Engineering and Technology, FEST, Hamdard University, Karachi, Pakistan.
Email: rizwan.tanweer@hamdard.edu.pk, muhammad170@e.ntu.edu.sg

[2]School of Computer Science and Engineering, Nanyang Technological University, Singapore, 639798.
Email: aldujail001@e.ntu.edu.sg, ssundaram@ntu.edu.sg

***Abstract*—These days, most of the real-world problems have become multi-criteria in nature and the demand for an effective multi-objective optimization algorithm has been significantly increased. This paper presents a new Multi-Objective Self-Regulating Particle Swarm Optimization (MOSRPSO) algorithm whereby the SRPSO algorithm originally developed for single objective problems has been modified to tackle with Multi-objective Optimization Problems (MOPs). The classical approach of Pareto dominance has been applied in the SRPSO framework together with the roulette wheel selection scheme for leader identification. The proposed MOSRPSO algorithm has been evaluated on all the hundred problems from Black-Box Multi-Objective Optimization Benchmarking (BMOBench) platform and the results are presented. The performance clearly indicate that MOSRPSO is a potential candidate for solving MOPs.**

***Index Terms*—MOO: Multi-Objective Optimization, PSO: Particle Swarm Optimization, MOSRPSO: Multi-Objective Self Regulating Particle Swarm Optimization, BMOBench: Black-Box Multi-Objective Optimization Benchmarking**

I. INTRODUCTION

Over the past decades, the complexity of the global optimization problems have significantly increased and has been extended to the multi-criteria optimization problems. In multi-criteria, the nature of optimization problems are extremely complex whereby several conflicting objectives are required to be handled simultaneously. Further, there is also a need of such an algorithm that can effectively solve the (Multi-Objective Optimization Problems (MOPs) with minimum computational budget. This has outstretched the demand of real parameter optimization schemes for solving real problems. Among the numerous algorithms introduced, the population-based optimization algorithms are effectively providing promising solutions with lesser computational requirements. There are several MOO algorithms developed in the literature starting from initially developed Pareto Archived Evolutionary Strategy (PAES) [1], Strength Pareto Evolutionary Algorithm (SPEA) [2], Non-dominated Sorting Genetic Algorithm (NSGA-II) [3], recently developed hybrid framework for MOPs [4], dividing rectangles [5] and Gap Optimized Multi-objective Optimization (GOMORS) [6].

One of the effective population based nature inspired optimization algorithm is the Particle Swarm Optimization (PSO) algorithm initially introduced in 1995 by Eberhart and Kennedy [7]. PSO is derived from the collective behavior of swarms in search of food. Members of the swarm are represented as particles in the algorithm and the food searched by birds is presented as potential solution. The particles fly in the search space updating the search patterns using their self and social experiences. Information sharing mechanism has been implemented in the algorithm to ensure proper movement towards the global optimum solution. Lesser computational efforts of PSO attracted researchers towards development of efficient MOO algorithm. Multi-Objective PSO (MOPSO) was introduced in [8], [9] whereby PSO was modified for solving MOPs.

A recent area in the development of PSO algorithms is derived from human thought processes and learning principles. Human beings are known to be intelligent and have good social cognizance [10]–[12]. Inspired from this, researchers developed several variants of PSO by incorporating different human learning principles. A Human cognition inspired PSO was introduced in [13]. Further different areas from human learning principles have been explored and incorporated in PSO framework for better convergence [14]–[19]. All the human learning principles inspired PSO variants have exhibited significant performance improvement in the convergence characteristics of the PSO algorithm. For a comprehensive empirical analysis on the performance of PSO variants, one can refer to [20]–[22].

In this paper, recently proposed variant of PSO, the Self Regulating Particle Swarm Optimization (SRPSO) algorithm has been modified to handle MOPs. The SRPSO algorithm is inspired from human self-learning principles where self-regulated inertia weight and self-perception based selection of direction from global best position strategies were incorporated. Utilizing these strategies, SRPSO has exhibited much better results and has provided faster convergence closer to the optimum solution. It has been stated in [16], [17] that SRPSO has exhibited competitive performances as compared to other PSO variants and other meta-heuristics. Therefore, the same algorithm has been modified for MOPs by incorporating the classical Pareto dominance scheme in the algorithms' framework. Further, a roulette wheel selection scheme has been incorporated for selection of leader particle. With the help of these schemes, the algorithm has become capable of solving multi-objective problems. The algorithm is referred to as

978-1-4799-7492-4/15/$31.00

Multi-Objective Self-Regulating Particle Swarm Optimization (MOSRPSO). MOSRPSO has been evaluated on all the hundred Black-Box Multi-Objective Optimization Benchmarking (BMOBench) problems. The performance of MOSRPSO indicate that the algorithm is capable of solving MOPs effectively.

The rest of the paper is organized as follows: Section II briefly describes the SRPSO algorithms. Section III presents the MOSRPSO algorithm. Section IV presents a detailed discussion on the experimental setup and performance of MOSRPSO in solving BMOBench problems. Section V summarizes the conclusions of the paper.

II. Brief Overview of the SRPSO algorithms

Self Regulating Particle Swarm Optimization (SRPSO) algorithm introduced in [19] is a new variant of PSO inspired from human self-learning principles where the best human learning strategies have been incorporated for solving single objective optimization problems. The two learning strategies proposed in SRPSO are, (i) the self-regulated inertia weight only for the best particle and (ii) self-perception for selection of direction from the global best position for the rest of the particles. Using these strategies, the particles search the optimum solution with accelerated exploration and intelligent exploitation. The self-regulated inertia weight strategy for the best particle is defined as:

$$\omega_i = \begin{cases} \omega_i(t) + \Delta\omega, & \text{for best particle} \\ \omega_i(t) - \Delta\omega, & \text{otherwise} \end{cases} \quad (1)$$

where $\omega_i(t)$ is the current inertia weight and $\Delta\omega$ is the change in inertia weight. There are two different velocity update equations, viz., one for the best particle and other for all the remaining particles. The velocity update equations are:

$$V_{id}^{t+1} = \omega_i V_{id}^t \ , \qquad (i = best) \quad (2)$$

$$V_{jd}^{t+1} = \omega_j V_{jd}^t + c_1 r_1 (P_{jd}^t - X_{jd}^t) + c_2 r_2 p_{jd}^{so} (P_{gd}^t - X_{jd}^t) \quad (3)$$

where $(j \neq i)$, V_{id} and X_{id} are the velocity and position respectively, of i^{th} particle in the d^{th} dimension. t and $t+1$ represents the current and next iteration respectively; P_{id}^t is the personal best for particle i in the d^{th} dimension and P_{gd}^t is the global best in d^{th} dimension. c_1 and c_2 represents the acceleration coefficients and r_1 and r_2 are the random numbers distributed uniformly within the range $[0, 1]$ and

$$p_{jd}^{so} = \begin{cases} 1, & \text{if } a > 0.5 \\ 0, & \text{otherwise} \end{cases} \quad (4)$$

where $a \sim U([0, 1])$ for all the directions of the global best position. The self-regulating inertia weight provided better exploration and self-perception based selection of global search direction intelligently exploits the search space. The strategies have provided faster convergence closer to the global optimum solution.

III. The Multi-Objective SRPSO algorithm

This section presents the modified MOSRPSO algorithm. First, the Pareto dominance scheme is described and then the MOSRPSO algorithm is presented.

A. Basic concepts of Pareto Dominance

In multi-objective search space, the concept of dominance is incorporated for finding the appropriate solution from the search space. During the search process, a Multi-Objective algorithm produces a set of solutions instead of a single solution as there there is a presence of conflicting objective functions. Based on this concept, if solution x_i is better than solution x_j then solution x_i is said to be the dominating one. Once the dominant solution is found, the Pareto optimality is applied to get a non-dominant set of solution. These non-dominated solutions are stored as particles' best positions.

B. The MOSRPSO Algorithm

As discussed in the previous section, the Pareto dominance scheme has been utilized in the SRPSO framework to determine the best position. The best position is determined as the leader particle. As proposed in the classical MOPSO algorithm [9], a similar strategy has been incorporated in the MOSRPSO algorithm. The non-dominated solutions identified by the swarm are first stored in an archive (repository) for future references. Further, the entire search space is divided into several hypercubes whereby each hypercube contains a set of particles. In each hypercube, there is a fitness value assigned as the objective value of the space. The particles in each hypercube, search the entire cubic space for the optimal objective value. Finally, a roulette wheel selection scheme has been utilized to determine the potential hypercube and leader from the same hypercube.

In the SRPSO framework, the best particle utilizes different learning strategy compared to that of all the other particles. Similarly, in MOSRPSO the leader (dominant particle) is observed with the same learning strategy. The leader will follow the self-regulating inertia weight strategy whereas all the other particles in each hypercube will perform search utilizing self-perception based selection strategy. Therefore, the velocity update equation for the leader is:

$$V_{id}^{t+1} = \omega_i V_{id}^t \ , \qquad (i = best) \quad (5)$$

and the velocity update equation for all the remaining particles is:

$$V_{jd}^{t+1} = \omega_j V_{jd}^t + c_1 r_1 (P_{jd}^t - X_{jd}^t) + c_2 r_2 p_{jd}^{so} (R_{jh}^t - X_{jd}^t) \quad (6)$$

where P_{jd} is the personal best position and R_{jh} is the selected leader from the repository.

The pseudo-code for MOSRPSO algorithm, incorporating the proposed schemes is summarized in Algorithm 1.

IV. Performance Evaluation, Results and Discussion

The MOSRPSO has been empirically assessed on the BMOBench platform according to [23], where each algorithm is run on 100 multi-objective problems categorized over seven groups: low-dimensional, high-dimensional, uni-modal, multi-modal, and mixed categories [23]. To validate the efficacy of the proposed—stochastic—algorithm, its performance is compared with that of recent stochastic multi-objective solvers,

Algorithm 1: The MOSRPSO Algorithm

```
Initialization:
for each particle i do
    Randomly initialize position of each particle X_i in
    the search range (X_min, X_max)
    Randomly initialize velocity of each particle V_i
    Initialize the external archive
    Set the quality as leader
end
while (success= 0 and t ≤ max_iterations) do
    for each particle do
        select a leader from the external archive
        Calculate the fitness values for each particle;
        if (fitness value of the objective is better than the
        best fitness value of the objective (pbest), then
            Current fitness value of the objective function
            is set as the new pbest
    end
    for the best particle do
        Calculate the inertia weight ω using equation (1);
        Update the velocity using equation (5);
    end
    for the remaining particles do
        for j = 1 : Dimension do
            Generate the uniform random number a;
            if (a > 0.5), then
                Select the directions from global best
            else
                Reject the directions
            end
        end
        Update the velocity using equation (6);
    end
    Update the position of each particle;
    Update leader in the external archive;
end
```

namely MO−HOO [24] and the stochastic variant of DMS [25], referred to here as sDMS.

The procedure for assessing the solution quality of an algorithm is based on recording its *runtime*: the number of function evaluations required by the algorithm for its solution to reach a specific (target) quality value. The recorded runtimes are then expressed in terms of data profiles, which capture various aspects of the algorithms' convergence behavior. For more details on the problems, their characteristics and the evaluation procedure, one can refer to [23].

A. Experimental Setup

The parameter settings for MOSRPSO are: ω = 1.05 *to* 0.5, c_1 = c_2 = 1.49445, swarm size (n) = 200, archive size = 100, $Vmax$ = 0.1×search range. The experiments are conducted using MATLAB R2013a on a PC with: 64-bit Windows 7, Intel Xeon E5 CPU @ 3.20GHz, 16GB of memory.

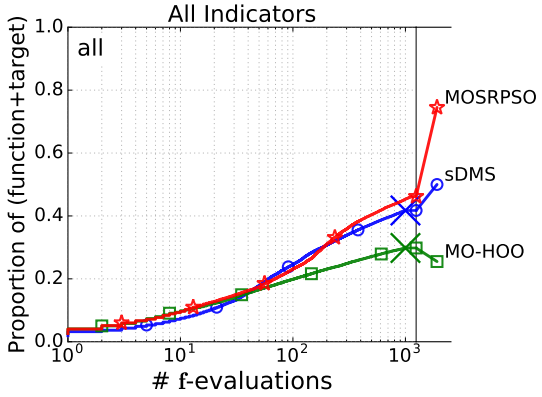

Fig. 1. Data profiles aggregated over all the problems across all the quality indicators computed for each of the compared algorithms. The symbol indicates the maximum number of function evaluations.

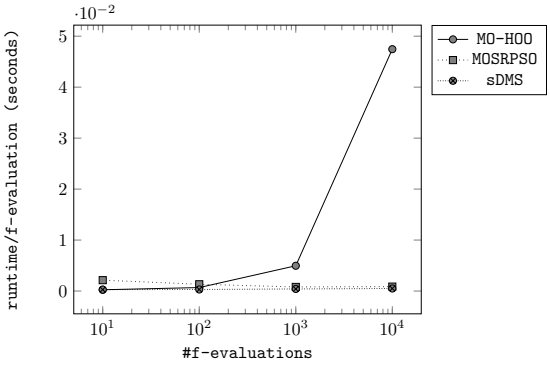

Fig. 2. A *semi*-log plot visualizing the runtime per one function evaluation (in seconds) of the compared algorithms. All the algorithms were run on a selected set of problems over a set of evaluation budgets, namely BK1, DPAM1, L3ZDT1, DTLZ3, and FES3; with an evaluation budget $\in \{10, 100, 1000, 10000\}$ per problem on a PC with: 64-bit Windows 7, Intel Xeon E5 CPU @ 3.20GHz, 16GB of memory.

B. Results and Discussion

Figures 3 and 4 show the data profiles of the compared algorithms as a function of the number of function evaluations used over different categories of the test problems according to [23] in terms of four commonly-used quality indicators, namely the hypervolume, the additive epsilon and the generational distance, the inverted generational distance, respectively. On the other hand, Figure 1 aggregates the data profiles of Figures 3 and 4 into a single plot, showing the overall performance of the compared algorithms.

From Figures 3 and 4, one can easily see that the performance of MOSRPSO is significantly better than others on the set of BMOBench platform problems. In fact, the performance of MOSRPSO is better in all categories of problems except the non-separable problems. It is indicated in the single objective SRPSO [19] and also in [17] that SRPSO is a rotationally variant algorithm which cannot perform efficient search on non-separable problems and the same is indicated in the data profiles presented in the figure. On all the other problems, the MOSRPSO algorithm has shown better performance compared

to sDMS and MO−HOO in terms of the hypervolume, additive epsilon, and inverted generational distance indicators. On the other hand, data profiles computed using the generational distance (GD) indicator favors sDMS. One can attribute this to the fact the GD indicator is sensitive to the size of the algorithm's approximation set—the bigger the set, the better the GD indicator value. A look into the size of sDMS and MOSRPSO's approximation sets confirms this observation: e.g., the size of the online archive for the Jin2 problem is 1197 and 51 for sDMS and MOSRPSO, respectively. Finally, from Figure 1 the overall performance of MOSRPSO can be observed as comparably better than other algorithms. This clearly indicates the MOSRPSO is a robust algorithm and a potential candidate for solving multi-objective problems efficiently and effectively.

C. Empirical Runtime Evaluation

In order to evaluate the complexity of the algorithms (measured in runtime), the algorithms are run on a representative set of the problems. The empirical complexity of an algorithm is then computed as the running time (in seconds) of the algorithm summed over all the problems given divided by the total number of function evaluations used. The results for four different evaluation budgets are shown in Figure 2. MO−HOO's computational complexity grows almost quadratically in the number of function evaluations due to the back-propagation applied on the nodes of the algorithm's tree and the identification of non-dominated fronts, which is dependent on the number of function evaluations.

V. CONCLUSIONS

This paper presents a new multi-objective self regulating particle swarm optimization algorithm. The SRPSO algorithm is modified utilizing the concept of Pareto dominance and roulette wheel selection scheme to cast the multi-objective problem solving scheme in the SRPSO framework. With the help of incorporated schemes, the MOSRPSO algorithm has successfully solved the multi-objective optimization problems. The performance has been evaluated on the BMOBench platform problems and the results indicate that MOSRPSO is capable of solving MOPs effectively. Therefore, MOSRPSO is a potential candidate for solving multi-objective optimization problems.

ACKNOWLEDGMENT

The authors wish to extend their thanks to the ATMRI:2014-R8, Singapore, for providing financial support to conduct this study.

REFERENCES

[1] J. Knowles and D. Corne, "The pareto archived evolution strategy: A new baseline algorithm for pareto multiobjective optimisation," in *IEEE Congress on Evolutionary Computation*, vol. 1. IEEE, 1999.

[2] E. Zitzler and L. Thiele, "An evolutionary algorithm for multiobjective optimization: The strength pareto approach," 1998.

[3] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation (CEC)*, vol. 6, no. 2, pp. 182–197, 2002.

[4] K. Sindhya, K. Miettinen, and K. Deb, "A hybrid framework for evolutionary multi-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 4, pp. 495–511, 2013.

[5] A. Al-Dujaili and S. Suresh, "Dividing rectangles attack multi-objective optimization," in *IEEE Congress on Evolutionary Computation (CEC)*, Vancouver, Canada, 2016.

[6] T. Akhtar and C. A. Shoemaker, "Multi objective optimization of computationally expensive multi-modal functions with RBF surrogates and multi-rule selection," *Journal of Global Optimization*, vol. 64, no. 1, pp. 17–32, 2015. [Online]. Available: http://dx.doi.org/10.1007/s10898-015-0270-y

[7] R. C. Eberhart, J. Kennedy *et al.*, "A new optimizer using particle swarm theory," in *Proceedings of the sixth international symposium on micro machine and human science*, vol. 1. New York, NY, 1995, pp. 39–43.

[8] C. A. C. Coello and M. S. Lechuga, "Mopso: a proposal for multiple objective particle swarm optimization," in *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on*, vol. 2, 2002, pp. 1051–1056.

[9] C. A. C. Coello, G. T. Pulido, and M. S. Lechuga, "Handling multiple objectives with particle swarm optimization," *IEEE Transactions on evolutionary computation*, vol. 8, no. 3, pp. 256–279, 2004.

[10] M. R. Tanweer, S. Suresh, and N. Sundararajan, "Human meta-cognition inspired collaborative search algorithm for optimization," in *Multisensor Fusion and Information Integration for Intelligent Systems (MFI), 2014 International Conference on*, Sept 2014, pp. 1–6.

[11] Y. Shi, "Brain storm optimization algorithm," in *International Conference in Swarm Intelligence*. Springer, 2011, pp. 303–309.

[12] A. Al-Dujaili, K. Subramanian, and S. Suresh, "Humancog: A cognitive architecture for solving optimization problems," in *2015 IEEE Congress on Evolutionary Computation (CEC)*, May 2015, pp. 3220–3227.

[13] M. R. Tanweer and S. Sundaram, "Human cognition inspired particle swarm optimization algorithm," in *Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2014 IEEE Ninth International Conference on*. IEEE, 2014, pp. 1–6.

[14] R. Cheng and Y. Jin, "A social learning particle swarm optimization algorithm for scalable optimization," *Information Sciences*, vol. 291, pp. 43–60, 2015.

[15] M. Tanweer, S. Suresh, and N. Sundararajan, "Mentoring based particle swarm optimization algorithm for faster convergence," in *2015 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2015, pp. 196–203.

[16] ——, "Dynamic mentoring and self-regulation based particle swarm optimization algorithm for solving complex real-world optimization problems," *Information Sciences*, vol. 326, pp. 1–24, 2016.

[17] M. Tanweer, R. Auditya, S. Suresh, N. Sundararajan, and N. Srikanth, "Directionally driven self-regulating particle swarm optimization algorithm," *Swarm and Evolutionary Computation*, vol. 28, pp. 98–116, 2016.

[18] M. R. Tanweer, S. Suresh, and N. Sundararajan, "Improved srpso algorithm for solving cec 2015 computationally expensive numerical optimization problems," in *2015 IEEE Congress on Evolutionary Computation (CEC)*, May 2015, pp. 1943–1949.

[19] M. Tanweer, S. Suresh, and N. Sundararajan, "Self regulating particle swarm optimization algorithm," *Information Sciences*, vol. 294, pp. 182–202, 2015.

[20] A. Al-Dujaili, M. R. Tanweer, and S. Suresh, "On the performance of particle swarm optimization algorithms in solving cheap problems," in *IEEE Symposium Series on Computational Intelligence*, Dec 2015, pp. 1318–1325.

[21] ——, "De vs. pso: A performance assessment for expensive problems," in *IEEE Symposium Series on Computational Intelligence*, Dec 2015, pp. 1711–1718.

[22] M. Tanweer, A. Al-Dujaili, and S. Suresh, "Empirical assessment of human learning principles inspired pso algorithms on continuous black-box optimization testbed," *International Conference on Swarm, Evolutionary and Memetic Computing (SEMCCO 2015)*, 2015.

[23] A. Al-Dujaili and S. Suresh, "BMOBench: Black-box multi-objective optimization benchmarking platform," *ArXiv e-prints*, vol. arXiv:1605.07009, 2016.

[24] K. Van Moffaert, K. Van Vaerenbergh, P. Vrancx, and A. Nowé, "Multi-objective χ-armed bandits," in *Neural Networks (IJCNN), 2014 International Joint Conference on*. IEEE, 2014, pp. 2331–2338.

[25] A. L. Custódio, J. A. Madeira, A. I. F. Vaz, and L. N. Vicente, "Direct multisearch for multiobjective optimization," *SIAM Journal on Optimization*, vol. 21, no. 3, pp. 1109–1140, 2011.

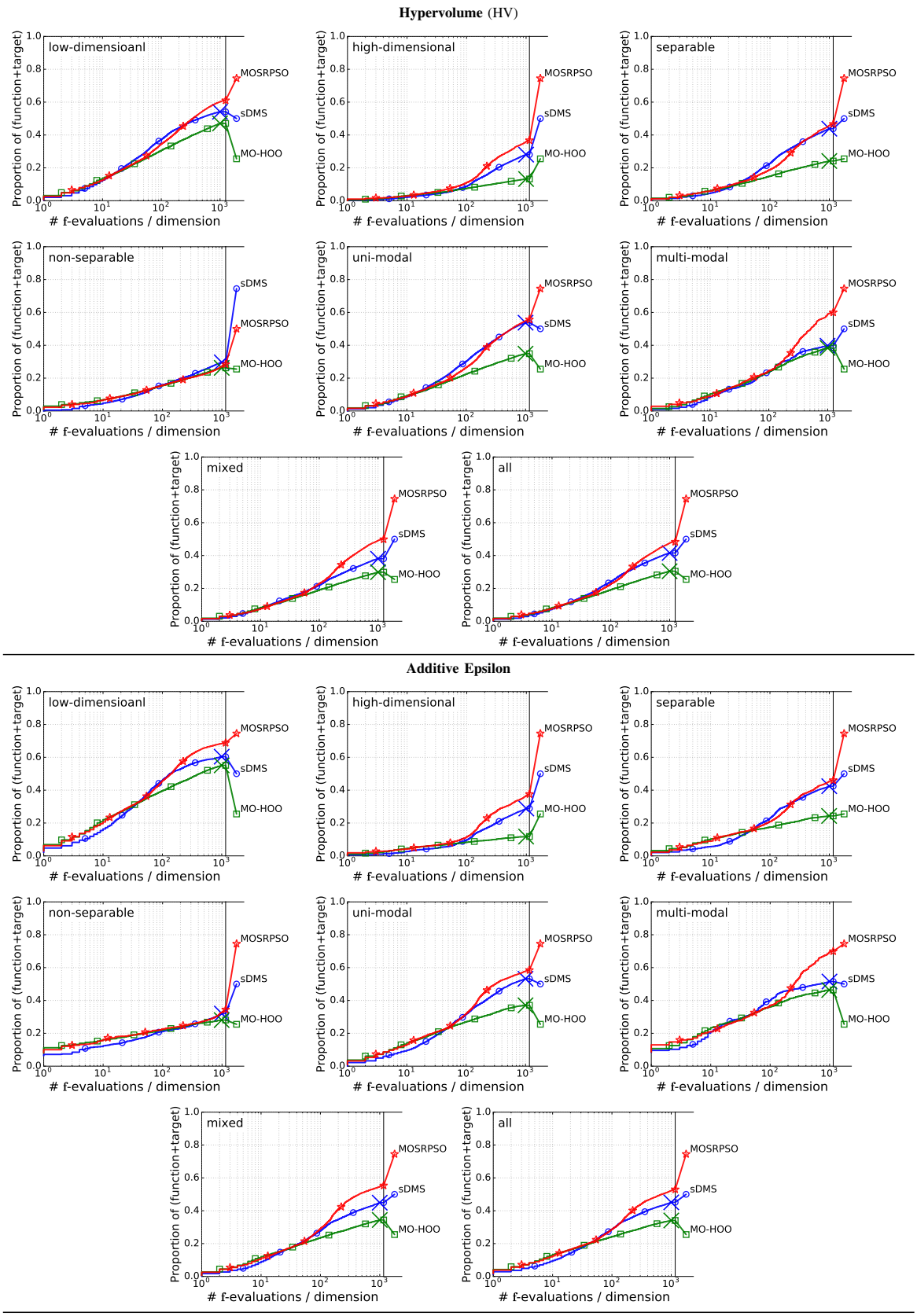

Fig. 3. Data profiles aggregated over problem categories for each of the quality indicators—the hypervolume and the additive epsilon. The symbol $\times$ indicates the maximum number of function evaluations. The y-axis represents the fraction of targets (a set of the considered indicator values) hit across the considered problems.

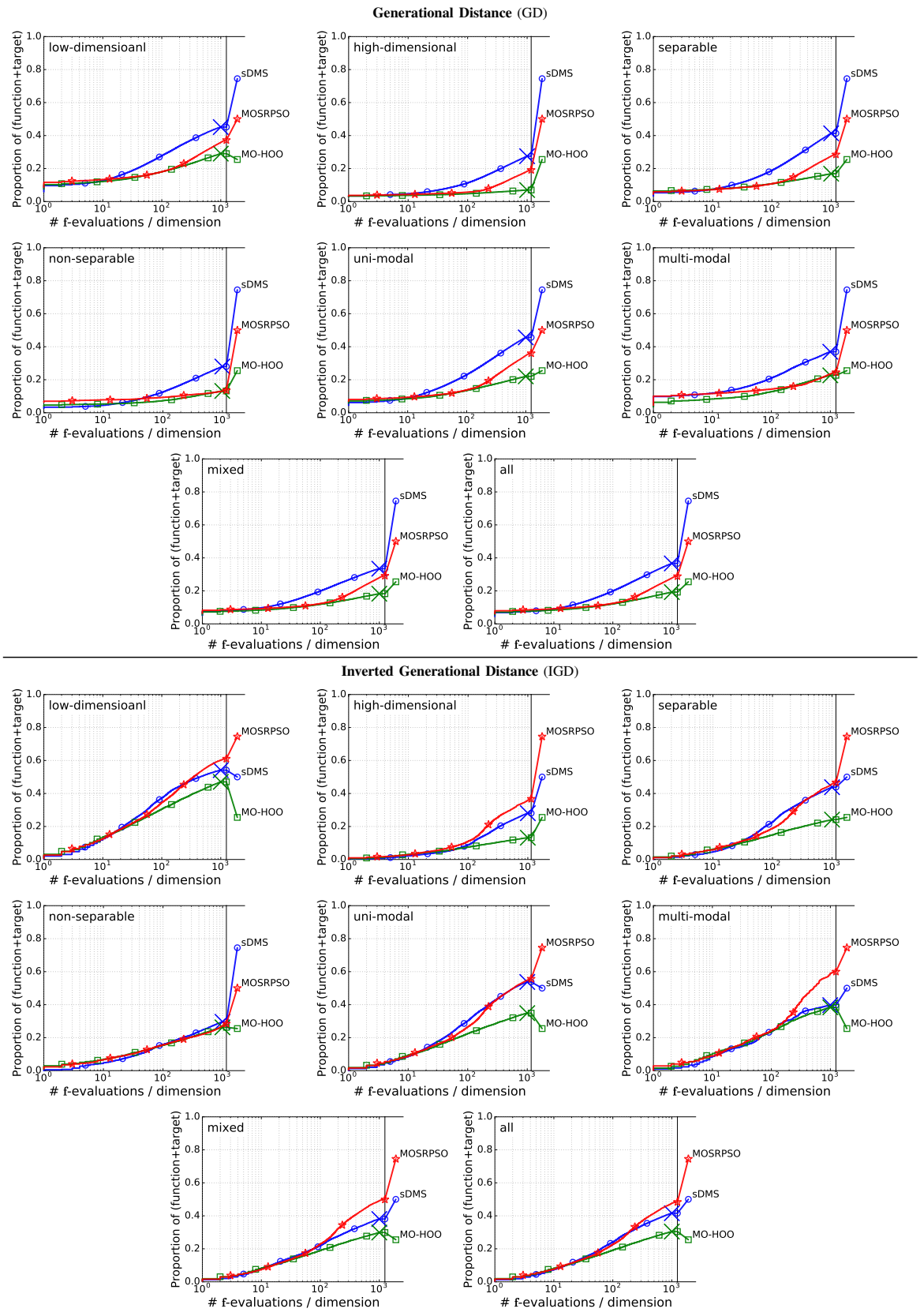

Fig. 4. Data profiles aggregated over problem categories for each of the quality indicators—the generational distance and the inverted generational distance. The symbol $\times$ indicates the maximum number of function evaluations. The y-axis represents the fraction of targets (a set of the considered indicator values) hit across the considered problems.