

MOEA/D with Random Partial Update Strategy

Yuri Lavinas*, Claus Aranha*, Marcelo Ladeira† and Felipe Campelo‡

*University of Tsukuba, School of Systems and Information Engineering

Email: lavinas.yuri.xp@alumni.tsukuba.ac.jp;

Email: caranha@cs.tsukuba.ac.jp

†University of Brasilia, Department of Computer Science

Email: mladeira@unb.br

‡Aston University, School of Engineering and Applied Sciences

Email: f.campelo@aston.ac.uk

Abstract—Recent studies on resource allocation suggest that some subproblems are more important than others in the context of the MOEA/D, and that focusing on the most relevant ones can consistently improve the performance of that algorithm. These studies share the common characteristic of updating only a fraction of the population at any given iteration of the algorithm. In this work, we investigate a new, more straightforward partial update strategy, in which a random subset of solutions is selected at every iteration. The performance of the MOEA/D-DE using this new resource allocation approach is compared experimentally against that of the standard MOEA/D-DE and the MOEA/D-DE with relative improvement-based resource allocation. The results indicate that using MOEA/D with this new partial update strategy results in improved HV and IGD values, and a much higher proportion of non-dominated solutions, particularly as the number of updated solutions at every iteration is reduced.

Index Terms—Multi-Objective Optimization, MOEA/D, Resource Allocation, Partial Update Strategy.

I. INTRODUCTION

Multi-objective Optimization Problems (MOPs) appear in many application contexts in which several conflicting objective functions need to be simultaneously optimized. Finding good sets of solutions for general continuous MOPs is generally considered a hard problem, mainly when convexity or differentiability cannot be assumed, for which Evolutionary Algorithms have been proposed as potential solvers [1]–[3].

The Multi-Objective Evolutionary Algorithm Based on Decomposition (MOEA/D) [4] is generally considered a practical algorithm for solving MOPs. The key idea of the MOEA/D is to decompose the multi-objective optimization problem into a set of single-objective subproblems, which are solved simultaneously by a population-based evolutionary approach.

While the original MOEA/D and some of its earlier variants did not discriminate between subproblems, it has since become clear that focusing computational effort on specific subsets of these subproblems can substantially improve the performance of the algorithm [5]–[9]. Also, it has been noted that the MOEA/D may sometimes waste computational effort by trying to improve solutions that are not very promising [10]. This waste of computational effort can be a critical issue, particularly in specific MOPs which require costly simulations to evaluate solutions [11]. Several works have proposed to address this issue and to investigate methods to allocate different

amounts of computational effort to subproblems, based on a variety of *priority functions* [5], [6], [8], [12], [13]. These approaches, which became collectively known as Resource Allocation (RA) techniques, have been shown to result in consistent performance improvements for the MOEA/D.

While different RA techniques have their particular characteristics, all share the characteristic feature of limiting the number of solutions from the population that is updated at any iteration. In a previous work [12], we observed the somewhat surprising result that is assigning random priority values to subproblems performed better than not using RA at all. In a similar result, Pruvost et. al [9] also found that selecting a subset of subproblems at random on MOEA/D performs well on the combinatorial domain. These results suggest that increasing the inertia of the population dynamics in the MOEA/D can be beneficial in itself, regardless of the Resource Allocation strategy.

The question that consequently arises can be summarized as: how much of the performance improvements observed in MOEA/Ds with Resource Allocation is due to the RA strategy itself, and how much can be attributed to the simple increase in the inertia of the population dynamics of the MOEA/D, which results from maintaining parts of the population unchanged between iterations? This work focuses on investigating and quantifying the extent of these effects. We also analyze which proportion of the population should be updated at any given iteration, to obtain improved performances for the MOEA/D. To investigate this effect, we introduce a Partial Update strategy, which allows us to control the proportion of the subproblems selected for variation at any iteration. We perform an experimental investigation of the impact of a proportion parameter ps on the performance of the MOEA/D on standard problem benchmarks.

The remainder of this paper is organized as follows: Section II reviews the main concepts related to resource allocation in the MOEA/D. Section III introduces the MOEA/D with Partial Update strategy. Sections IV and V present experimental results related to the investigation of the effect of partial updates on the performance of the algorithm, as well as comparisons against a baseline algorithm and an existing MOEA/D with resource allocation, using the Relative Improvement priority function. Finally, section VI presents our concluding remarks.

II. RESOURCE ALLOCATION

The key idea behind MOEA/D is to decompose a MOP into a set of single-objective subproblems, which are then solved simultaneously. While these subproblems are usually considered equivalent, a growing body of work indicates that prioritizing some subproblems at specific points of the search can improve the performance of MOEA/D. This issue is commonly addressed using resource allocation (RA) techniques.

Priority functions are used in resource allocation to determine preferences between subproblems. These functions take information about the progress of the search and return priority values that are then used to change the distribution of computational resources among subproblems at any given iteration [14]. They also allow the design of MOEA/D variants that allocate more resources on any desired solution characteristics [12], such as diversity or robustness [15].

Priority functions mediates the distribution of computational resources using a thresholding operation. At any given iteration t , let u_i^t indicate the priority function value attributed to the i -th subproblem, and v^t be a threshold value. The subset of solutions selected for variation on that iteration is then defined as the subproblems for which $u_i^t \geq v^t$.

The original work on resource allocation for the MOEA/D [5] defined a priority function known as the Relative Improvement (RI), defined as:

$$u_i^t = \frac{f(\mathbf{x}_i^{t-\Delta t}) - f(\mathbf{x}_i^t)}{f(\mathbf{x}_i^{t-\Delta t})}, \quad (1)$$

where $f(\mathbf{x}_i^t)$ represents the aggregation function value of the incumbent solution to the i -th subproblem on iteration t . Δt is a parameter that controls how many generations to wait for the relative improvement comparison (notice that this definition assumes a minimization problem and an aggregation function that always yields strictly positive values).

A. State of the art

Much of the research on resource allocation have used RI as a priority function, with some modifications on other aspects of the algorithm [6], [16]. Zhou et al. did expand the discussion over resource allocation in their work [6]. However, few other works have studied resource allocation in depth.

For example, both MOEA/D-GRA and MOEA/D-DRA use the RI priority function. That said, MOEA/D-GRA [16] uses a different replacement strategy to avoid newly generated solutions from updating several neighboring subproblems at any iteration. on the other hand, MOEA/D-DRA [5] performs a more sophisticated procedure for selecting subproblems using a 10-tournament selection based on the RI priority values.

Two works that attempted to investigate distinct priority functions are the EAG-MOEA/D [14] and MOEA/D-CRA [7]. Both used priority functions which allocate resources according to the possibility that subproblems may either be improved or contribute to the improvement of other subproblems.

In previous works, we isolated the priority function as a point of investigation [8], [12]. The goal in these works was

to improve the performance of MOEA/D based on the choice of priority function and to understand further the behavior of MOEA/D under different resource allocation approaches. On these works, we introduced two new priority functions (DS and iDS), based on the conjecture that MOEA/D would benefit from a greater focus on diversity in the decision space. Experimental comparisons were performed between the MOEA/D-DE under three priority functions: (1) RI, (2) DS and iDS; and under two methods used as a baseline: (1) MOEA/D-DE using randomly assigned priority values and (2) MOEA/D-DE without any resource allocation method.

These experiments revealed the somewhat surprising result that is using a random resource allocation performed as well as RI, and better than not using resource allocation at all. This result suggests that MOEA/D may benefit simply from the increased populational inertia (possibly due to slower diversity loss) that results from holding portions of the population constant during any given iteration.

To further investigate this question, we propose using an update strategy for the MOEA/D based on randomly allocated priority values. This strategy allows us to control the expected number of subproblems modified at any given iteration and, consequently, to (partially) regulate the population dynamics of the MOEA/D. This approach is described in the next section.

III. A NEW UPDATE STRATEGY FOR MOEA/D

To verify whether there is a positive effect in limiting the number of solutions updated at each iteration, and to investigate the extent of this effect, we introduce the Partial Update strategy. This strategy defines the expected amount of solutions updated at each iteration, regulated by a control parameter, $ps \in (0, 1]$. This parameter represents the probability that a given subproblem will be selected for updating at a given iteration. Notice that, under this definition, the allocation of resources to subproblems is entirely random, and any effects observed on the performance of the MOEA/D under this allocation strategy will be due only to the impact of maintaining portions of the population unchanged across iterations and their influence with each other. Algorithm 1 details the pseudocode of the MOEA/D-DE using the Partial Update Strategy.

Notice that the standard MOEA/D, as well as variants such as MOEA/D-DE [5], can be instantiated from Algorithm 1 by setting $ps = 1$. The only difference that the partial update strategy introduces in the base algorithm is that only a few subproblems are updated (probabilistically) at any given iteration, regulated by the value of ps .

Also, MOEA/D-PS maintains the Δt parameter from RI. Since MOEA/D-PS does not have an explicit priority function, this parameter just makes the algorithm work in two phases during the search progress. In the first phase, all subproblems are updated at every iteration, i.e., with no difference from the usual MOEA/D-DE approach. This initial phase lasts for Δt iterations. After that, the algorithm moves onto the

Algorithm 1 MOEA/D-PS (MOEA/D-DE with Partial Update Strategy)

```
1: Input:  $ps$ ,  $\Delta t$ , Termination criteria, MOEA/D-DE parameters.
2: Initialize MOEA/D-DE variables (e.g. weight vectors, set of solutions, etc.)
3:  $t \leftarrow 0$ 
4:  $u_i \leftarrow 1$ 
5: while Termination criteria do
6:    $t \leftarrow t + 1$ 
7:   if  $t \geq \Delta t$  then
8:      $u_i \leftarrow ps$        $\triangleright$  Allocation of update probability
9:   end if
10:  for  $i = 1$  to  $N$  do       $\triangleright$  Number of subproblems
11:    if  $\text{rand}() < u_i$  then
12:      Generate new candidate  $y$  for subproblem  $i$ .
13:    end if
14:    Update the set of solutions by  $y$ .
15:  end for
16:  Evaluate the set of solutions.
17: end while
```

second phase, during which MOEA/D-PS performs (randomly selected) partial updates at every iteration.

With this structure, other resource allocation techniques could also be expressed, by modifying the priority value attribution function in Line 8 of the algorithm (and possibly setting Δt to zero, if the initial phase is not desired).

It is relevant to observe that subproblems that are not selected by the partial update strategy at a given iteration may still have their incumbent solutions updated. Resource allocation in MOEA/D-PS affects only the variation step, not the replacement one; thus, subproblems not selected for variation may receive new candidate solutions, e.g., generated for a neighboring subproblem.

IV. PARTIAL UPDATE STRATEGY PARAMETER STUDY

To isolate and examine the effects of updating only part of the MOEA/D-DE population at any iteration, we performed a comparative experiment using two known benchmark sets. Six update levels were used, with $ps \in \{0.1, 0.2, 0.4, 0.6, 0.8, 1.0\}$ (the last of which simply selects all subproblems for updating at every iteration, and represents the standard algorithm without any resource allocation strategy). The MOEA/D-DE implementation from the MOEADr package [17], [18] was used as a basis, with modifications included to enable the use of the Partial Update technique as described in the previous section. Note that when the ps parameter is equal to 1.0, all subproblems are selected to be updated; therefore, this case simply reproduces the standard MOEA/D-DE.

A. Benchmark Problems

Two benchmark sets were used: the scalable DTLZ set [19], with 2 objectives, and the UF set [20]. In both cases we used

the test functions with dimension $D = 100$. The implementation of the test problems available from the *smoof* package [21] was used in all experiments.

The DTLZ Benchmark set is composed of seven unconstrained test problems, with distinct problem features [22]:

- DTLZ1: Linear Pareto Front - unimodal;
- DTLZ2: Concave Pareto Front - unimodal;
- DTLZ3: Concave Pareto Front - multimodal;
- DTLZ4: Concave Pareto Front - unimodal;
- DTLZ5: Degenerate Pareto Front - unimodal;
- DTLZ6: Degenerate Pareto Front - unimodal;
- DTLZ7: Disconnected Pareto Front with concave and convex portions - multimodal.

The UF Benchmark set is composed of ten unconstrained test problems with Pareto sets designed to be challenging to existing algorithms [23]. Problems UF1-UF7 are two-objective MOPs, while UF8-UF10 are three-objective problems [20].

- UF1: Convex Pareto Front - multimodal;
- UF2: Convex Pareto Front - multimodal;
- UF3: Convex Pareto Front - multimodal;
- UF4: Concave Pareto Front - multimodal;
- UF5: Linear Pareto Front - multimodal;
- UF6: Linear Pareto Front - multimodal;
- UF7: Linear Pareto Front - multimodal;
- UF8: Concave Pareto Front - multimodal;
- UF9: Linear and discontinuous Pareto Front - multimodal;
- UF10: Concave Pareto Front - multimodal;

B. Experimental Parameters

We used the MOEA/D-DE parameters as they were introduced in the work of Li and Zhang [2] in all tests. Table I summarizes the experimental parameters. Regarding the Δt parameter, we use the value suggested by Zhou et al. [6]. We make this parameter choices to isolate the contribution of the ps parameter, which controls the proportion of subproblems updated, comparing this change directly with the original algorithms.

Details of these parameters can be found in the documentation of package MOEADr, as well as in the original MOEA/D-DE reference [5], [17], [18]. All objectives were linearly scaled at every iteration to the interval $[0, 1]$, and the Weighted Tchebycheff scalarization function was used.

C. Experimental Evaluation

We compare the results of the different strategies using the Hypervolume (HV, higher is better) and Inverted Generational Distance (IGD, lower is better) indicators. We also evaluate the proportion of non-dominated solutions in the final population. The differences among the techniques are analyzed using Wilcoxon Rank Sum Tests (all-vs-all), with a significance level of $\alpha = 0.05$ and Hommel adjustment for multiple

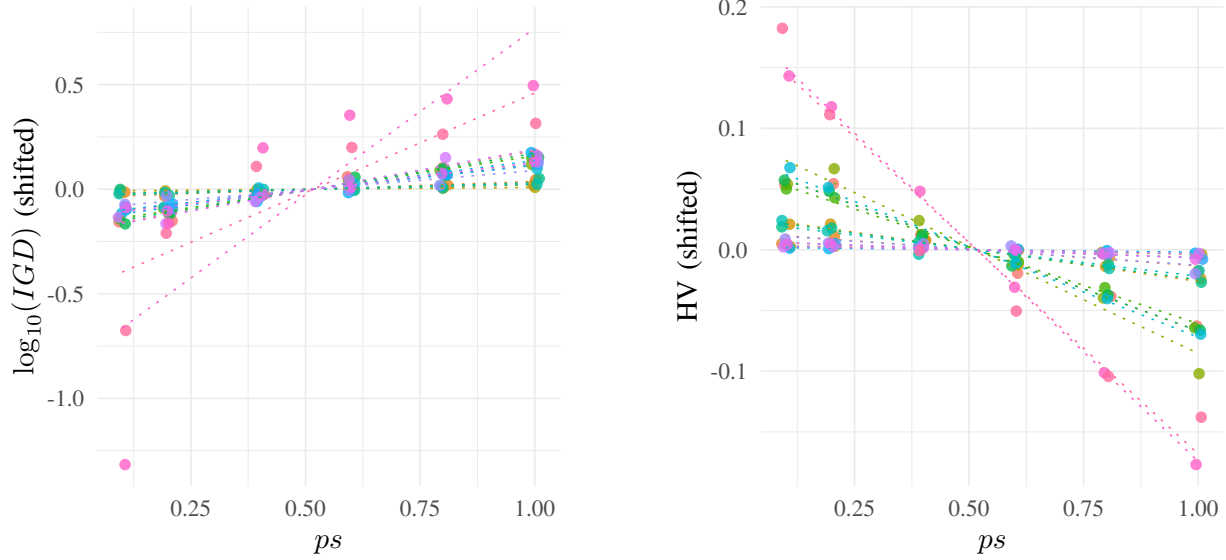


Fig. 1: Linear regression of IGD (lower is better, left) and HV (higher is better, right) against the ps parameter values. Each line represents an individual problem. It is clear that lower values of ps (smaller proportion of updated subproblems) are associated with better performance.

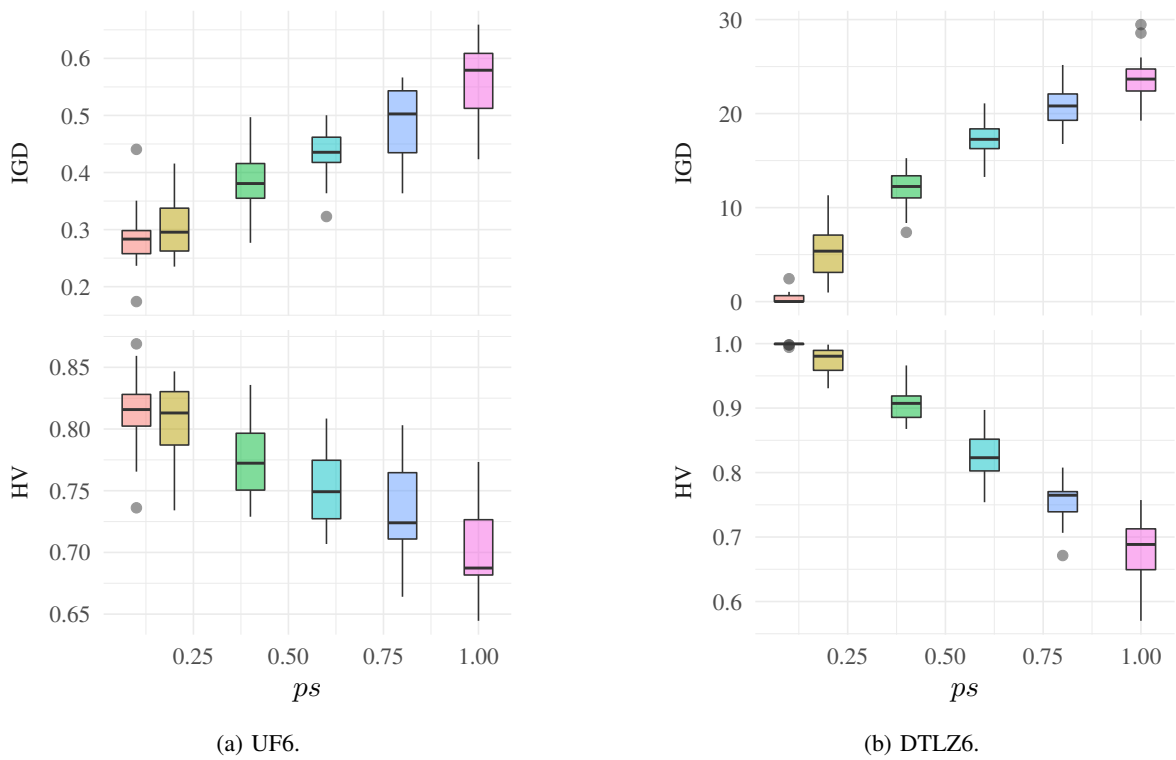


Fig. 2: Examples of distribution of IGD values (lower is better) and HV values (higher is better) for the final population, according to ps . Problems UF6 on the left and DTLZ6 on the right.

TABLE I: Experimental parameter settings.

MOEA/D-DE parameters	Value
DE mutation param.	$F = 0.25$
Polynomial mutation params.	$\eta_m = 20$ $p_m = 0.01$
Restricted Update param.	$nr = 2$
Locality parameter	$\delta_p = 0.9$
Neighborhood size	$T = 20$
SLD decomposition param.	$h = 349$ (2 obj) $h = 25$ (3 obj)
Population size	$N = 350$ (2 obj) $N = 351$ (3 obj)

Resource Allocation Parameters	Value
Generations before RA starts	$\Delta t = 20$

Experiment Parameters	Value
Repeated runs	21
Computational budget	30000 evals.

comparisons. For reproducibility purposes, all the code and experimental scripts are available online ¹.

For the calculation of HV, the objective function was scaled to the (0, 1) interval, with reference points set to (1, 1), for two-objective problems; and (1, 1, 1), for three-objective ones.

D. Results

Figure 1 shows regression lines of performance on ps for each test problem, both for log-IGD and HV (higher is better).² These results suggest a clear association between lower values of ps and improved performance on both indicators. Figure 2 provides a closer visualization of this effect in the case of two test problems, UF6 and DTLZ6, respectively. Statistical tests corroborate these observations, as summarised in Table II. The final raw data and analysis scripts can be retrieved from the project repository on Github.¹

E. Anytime Performance of MOEA/D-DE with Partial Update Strategy

Besides providing good final results, it is often desired that an MOEA be capable of returning a set of reasonably good solutions if interrupted at any time during the search [24], [25]. To investigate the impact of using distinct ps values on the anytime performance of the MOEA/D-DE with Partial Update we analyzed the effects in terms of both IGD and HV values.

Figures 3 illustrates the anytime performance of the MOEA/D-DE with Partial Update Strategy in terms of hypervolume (higher is better) for two specific problems while

¹<https://github.com/yclavinas/MOEADr/tree/cec2020>

²The log transformation was used to account for the large differences of scale in the IGD indicator due to DTLZ1 and DTLZ3, in which all configurations failed to converge adequately. It is possible that the computational budget of 30,000 candidate solution evaluations may not be enough for solving these two problems.

TABLE II: Statistical significance of differences in median IGD and HV, associated with different ps values. Values are Hommel-adjusted p-values of Wilcoxon Rank-sum tests. “ \uparrow ” indicates superiority of the column method, and “ \approx ” indicates differences not statistically significant (95% confidence level).

IGD					
ps	10%	20%	40%	60%	80%
20%	0.098 \approx				
40%	6.4e-4 \uparrow	0.006 \uparrow			
60%	1.8e-4 \uparrow	7.6e-5 \uparrow	7.6e-5 \uparrow		
80%	7.6e-5 \uparrow	7.6e-5 \uparrow	7.6e-5 \uparrow	7.6e-5 \uparrow	
100%	7.6e-5 \uparrow	7.6e-5 \uparrow	7.6e-5 \uparrow	7.6e-5 \uparrow	7.6e-5 \uparrow

HV					
ps	10%	20%	40%	60%	80%
20%	0.185 \approx				
40%	0.002 \uparrow	9.6e-4 \uparrow			
60%	9.6e-4 \uparrow	9.6e-4 \uparrow	0.001 \uparrow		
80%	9.6e-4 \uparrow	9.6e-4 \uparrow	9.6e-4 \uparrow	9.6e-4 \uparrow	
100%	9.6e-4 \uparrow	9.6e-4 \uparrow	9.6e-4 \uparrow	9.6e-4 \uparrow	9.6e-4 \uparrow

Figure 4 illustrates the anytime performance of the MOEA/D-DE with Partial Update Strategy in terms of IGD (lower is better). Please recall that all subproblems are selected regardless of ps until iteration $\Delta t = 20$ (see Section III for details). Consistently with the end-of-run results, Figures 3 and 4 indicate that changing smaller percentages of the population at each iteration tends to result in better performance anytime during the search. While this is only illustrated here for two test problems, the same behavior is observed for almost all other problems. We consider these results as an indication that smaller values of the ps parameter result in faster and better convergence for the MOEA/D-DE, at least for 2- and 3-objective problems with characteristics similar to the test ones employed in these experiments.

V. COMPARISON STUDY

In the previous section, we investigated the influence of different values of the control parameter ps on the performance of the MOEA/D-PS. The results indicate that low ps values are associated with (anytime) improvements in IGD and HV values. In this section, we compare the MOEA/D-PS (using $ps = 0.1$) against the original MOEA/D-DE and an MOEA/D-DE with resource allocation based on RI [5], [6]. The same test problems described in the previous section were used.

Table III tabulates the mean results obtained by the MOEA/D-PS with $ps = 0.1$, the pure MOEA/D-DE and the MOEA/D-DE with RI-based resource allocation, for all test problems. It is clear that the MOEA/D-PS results are considerably better when compared to the other methods, not only in terms of IGD and HV but also on the mean proportion of nondominated solutions (NDOM) that it returns in the final population. Table IV presents the results of statistical pairwise comparisons using the same methodology described in subsection IV-C, corroborating the results observed in Table III.

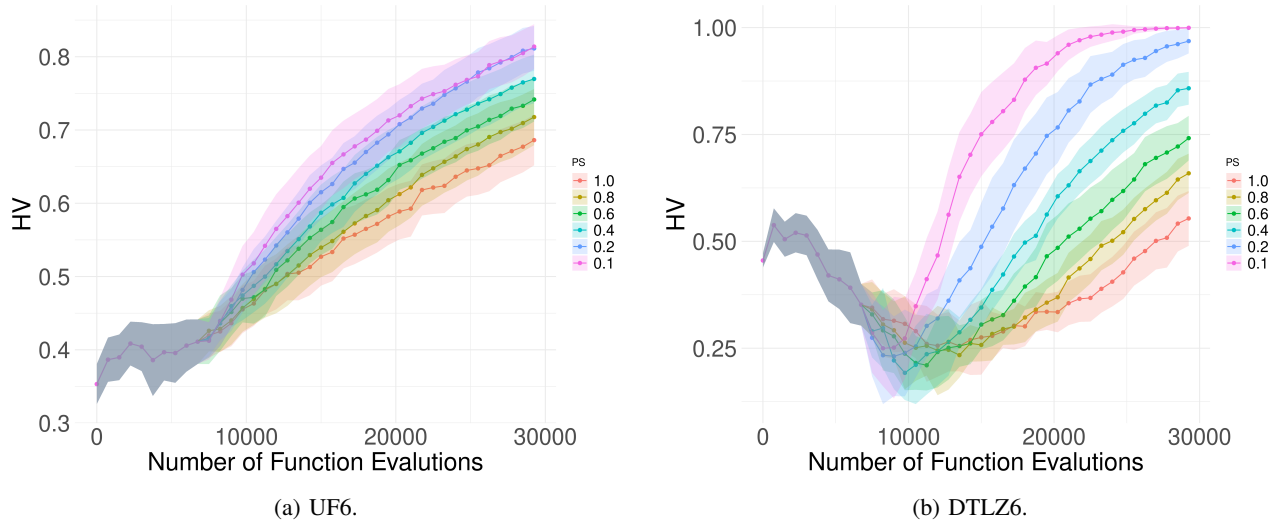


Fig. 3: Anytime HV (higher is better) performance of MOEA/D-PS for different values of ps on two functions.

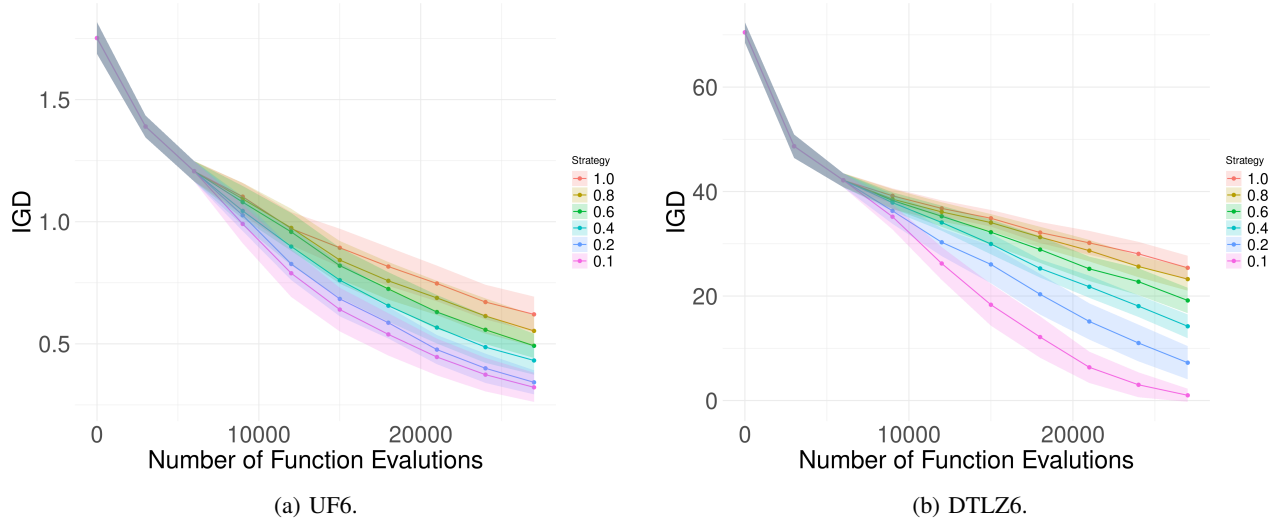


Fig. 4: Anytime IGD (lower is better) performance of MOEA/D-PS for different values of ps on two functions.

Looking at the proportion of non-dominated solutions (NDOM) in Table III, we see that randomly updating a small fraction of the subproblems at each iteration resulted in the highest value on all functions, often with a substantial lead.

In our view, a higher proportion of non-dominated solutions suggests a better, more diverse set of solutions in the objective space. This suggestion would indicate the use of the partial update strategy (under a low ps value) as a compelling strategy for improving convergence (subsection IV-E) and diversity in the MOEA/D.

VI. CONCLUSION

In this work, we presented a random partial update strategy for the MOEA/D-DE, which was incorporated into a simple algorithm (MOEA/D-PS). The partial update strategy adds one control parameter (ps), which regulates the proportion of the population that is selected for variation at any iteration.

Six ps values were investigated experimentally, revealing a strong association between more conservative updating of the MOEA/D-DE population (i.e., lower ps values) and improved performance. Based on these experiments, we suggest using small ps values, such as $ps = 0.1$. However, more thorough sensitivity analyses should be conducted to refine our understanding of these effects.

Besides, we showed that the MOEA/D-PS with $ps = 0.1$ values was able to outperform the pure MOEA/D-DE as well as a resource allocation MOEA/D-DE based on the well-know RI priority function. This result suggests that the MOEA/D-DE benefits more from having slower population dynamics than from a specific prioritization of subproblems based on the relative improvement criteria.

This study raises two issues that we consider essential for further explaining the effect of Partial Update strategies. The first is whether MOEA/D-PS would benefit from adapting the

TABLE III: Means and standard errors for IGD, HV and proportion of nondominated solutions (NDOM), for each algorithm-problem pair. The best point estimate for each problem is highlighted.

IGD			
	MOEA/D-PS	MOEA/D-DE	RI
UF1	0.26 ± 0.002	0.55 ± 0.003	0.37 ± 0.002
UF2	0.1 ± 0.001	0.12 ± 0.001	0.096 ± 0.001
UF3	0.28 ± 0.001	0.31 ± 0.001	0.29 ± 0.001
UF4	0.11 ± 0.001	0.11 ± 0.001	0.11 ± 0.001
UF5	1.1 ± 0.005	1.7 ± 0.003	1.3 ± 0.004
UF6	0.29 ± 0.003	0.56 ± 0.003	0.38 ± 0.003
UF7	0.26 ± 0.002	0.53 ± 0.003	0.36 ± 0.003
UF8	0.27 ± 0.001	0.31 ± 0.001	0.3 ± 0.001
UF9	0.42 ± 0.001	0.47 ± 0.001	0.46 ± 0.001
UF10	2.1 ± 0.023	3.4 ± 0.008	2.3 ± 0.012
DTLZ1	230 ± 5.5	440 ± 6.3	260 ± 5.7
DTLZ2	0.11 ± 0.001	0.18 ± 0.001	0.13 ± 0.001
DTLZ3	610 ± 16	970 ± 19	660 ± 17
DTLZ4	0.12 ± 0.002	0.23 ± 0.002	0.18 ± 0.006
DTLZ5	0.11 ± 0.001	0.19 ± 0.001	0.13 ± 0.001
DTLZ6	0.37 ± 0.03	24 ± 0.11	13 ± 0.14
DTLZ7	0.4 ± 0.013	3.9 ± 0.013	1.9 ± 0.034

HV			
	MOEA/D-PS	MOEA/D-DE	RI
UF1	0.86 ± 0.001	0.74 ± 0.001	0.82 ± 0.001
UF2	0.79 ± 0.001	0.76 ± 0.001	0.81 ± 0.001
UF3	0.57 ± 0.003	0.52 ± 0.001	0.55 ± 0.002
UF4	0.37 ± 0.001	0.37 ± 0.001	0.37 ± 0.001
UF5	0.72 ± 0.002	0.57 ± 0.001	0.69 ± 0.001
UF6	0.81 ± 0.001	0.7 ± 0.002	0.78 ± 0.001
UF7	0.83 ± 0.001	0.71 ± 0.001	0.79 ± 0.001
UF8	0.85 ± 0.001	0.81 ± 0.001	0.85 ± 0.001
UF9	0.78 ± 0.002	0.73 ± 0.001	0.74 ± 0.001
UF10	0.81 ± 0.003	0.67 ± 0.001	0.8 ± 0.002
DTLZ1	1 ± 6.5e - 5	0.99 ± 9.2e - 5	1 ± 6.9e - 5
DTLZ2	0.92 ± 0.001	0.91 ± 0.001	0.92 ± 0.001
DTLZ3	0.98 ± 0.001	0.96 ± 0.001	0.98 ± 0.001
DTLZ4	0.98 ± 5.6e - 5	0.97 ± 7.2e - 5	0.97 ± 0.001
DTLZ5	0.92 ± 0.001	0.91 ± 0.001	0.92 ± 0.001
DTLZ6	1 ± 5.8e - 5	0.68 ± 0.002	0.89 ± 0.001
DTLZ7	0.88 ± 0.001	0.56 ± 0.001	0.75 ± 0.002

NDOM			
	MOEA/D-PS	MOEA/D-DE	RI
UF1	0.89 ± 0.003	0.27 ± 0.002	0.45 ± 0.005
UF2	0.96 ± 0.002	0.42 ± 0.003	0.7 ± 0.008
UF3	0.92 ± 0.002	0.23 ± 0.002	0.43 ± 0.005
UF4	0.9 ± 0.003	0.68 ± 0.003	0.81 ± 0.004
UF5	0.85 ± 0.005	0.19 ± 0.001	0.43 ± 0.005
UF6	0.86 ± 0.003	0.29 ± 0.002	0.44 ± 0.004
UF7	0.92 ± 0.002	0.31 ± 0.002	0.5 ± 0.004
UF8	0.99 ± 0.001	0.54 ± 0.004	0.94 ± 0.002
UF9	0.99 ± 0.001	0.55 ± 0.003	0.88 ± 0.004
UF10	0.94 ± 0.003	0.44 ± 0.004	0.87 ± 0.004
DTLZ1	0.93 ± 0.004	0.1 ± 0.002	0.51 ± 0.01
DTLZ2	0.96 ± 0.002	0.3 ± 0.002	0.69 ± 0.01
DTLZ3	0.75 ± 0.01	0.046 ± 0.001	0.19 ± 0.006
DTLZ4	0.74 ± 0.006	0.18 ± 0.002	0.51 ± 0.009
DTLZ5	0.96 ± 0.002	0.3 ± 0.002	0.69 ± 0.009
DTLZ6	0.91 ± 0.009	0.063 ± 0.002	0.15 ± 0.004
DTLZ7	0.84 ± 0.005	0.22 ± 0.003	0.45 ± 0.012

TABLE IV: Statistical significance of differences in median IGD, HV and NDOM, for the three algorithms tested in this section. Values are Hommel-adjusted p-values of Wilcoxon Rank-sum tests. “↑” indicates superiority of the column method, and “≈” indicates differences not statistically significant (95% confidence level).

IGD		
	MOEA/D-PS	MOEA/D-RI
MOEA/D-RI	2.1e-4 ↑	
MOEA/D-DE	3.1e-5 ↑	3.1e-5 ↑

HV		
	MOEA/D-PS	MOEA/D-RI
MOEA/D-RI	6.4e-4 ↑	
MOEA/D-DE	0.0025 ↑	6.4e-4 ↑

NDOM		
	MOEA/D-PS	MOEA/D-RI
MOEA/D-RI	3.2e-4 ↑	
MOEA/D-DE	3.2e-4 ↑	3.2e-4 ↑

ps value throughout the search, either using a fixed schedule or through online adaptation. The second is the interaction effects between the ps value and other components of MOEA/D, such as decomposition strategy, neighborhood strategies, and other parameters of the algorithm. These would be interesting questions for further investigation.

REFERENCES

- [1] A. Trivedi, D. Srinivasan, K. Sanyal, and A. Ghosh, “A survey of multiobjective evolutionary algorithms based on decomposition,” *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 3, pp. 440–462, 2017.
- [2] H. Li and Q. Zhang, “Multiobjective optimization problems with complicated pareto sets, moea/d and nsga-ii,” *IEEE Transactions on evolutionary computation*, vol. 13, no. 2, pp. 284–302, 2009.
- [3] E. Zitzler and S. Künzli, “Indicator-based selection in multiobjective search,” in *International Conference on Parallel Problem Solving from Nature*. Springer, 2004, pp. 832–842.
- [4] Q. Zhang and H. Li, “Moea/d: A multiobjective evolutionary algorithm based on decomposition,” *IEEE Transactions on evolutionary computation*, vol. 11, no. 6, pp. 712–731, 2007.
- [5] Q. Zhang, W. Liu, and H. Li, “The performance of a new version of moea/d on cec09 unconstrained mop test instances,” in *Evolutionary Computation, 2009. CEC’09. IEEE Congress on*. IEEE, 2009, pp. 203–208.
- [6] A. Zhou and Q. Zhang, “Are all the subproblems equally important? resource allocation in decomposition-based multiobjective evolutionary algorithms,” *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 1, pp. 52–64, 2016.
- [7] Q. Kang, X. Song, M. Zhou, and L. Li, “A collaborative resource allocation strategy for decomposition-based multiobjective evolutionary algorithms,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2018.
- [8] Y. Lavinias, C. Aranha, and T. Sakurai, “Using diversity as a priority function for resource allocation on moea/d,” in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM, 2019, pp. 215–216.
- [9] G. Pruvost, B. Derbel, A. Liefvooghe, K. Li, and Q. Zhang, “On the combined impact of population size and sub-problem selection in moea/d,” in *European Conference on Evolutionary Computation in Combinatorial Optimization*. Springer, 2020, p. To Appear.
- [10] L. C. Bezerra, M. López-Ibáñez, and T. Stützle, “Comparing decomposition-based and automatically component-wise designed multi-objective evolutionary algorithms,” in *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, 2015, pp. 396–410.

- [11] T. Kohira, H. Kemmotsu, O. Akira, and T. Tatsukawa, "Proposal of benchmark problem based on real-world car structure design optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM, 2018, pp. 183–184.
- [12] Y. Lavinias, C. Aranha, and M. Ladeira, "Improving resource allocation in moea/d with decision-space diversity metrics," in *Theory and Practice of Natural Computing*. Cham: Springer International Publishing, 2019, pp. 134–146.
- [13] P. Wang, W. Zhu, H. Liu, B. Liao, L. Cai, X. Wei, S. Ren, and J. Yang, "A new resource allocation strategy based on the relationship between subproblems for moea/d," *Information Sciences*, 2019.
- [14] X. Cai, Y. Li, Z. Fan, and Q. Zhang, "An external archive guided multi-objective evolutionary algorithm based on decomposition for combinatorial optimization," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 4, pp. 508–523, 2015.
- [15] F. Goulart, S. T. Borges, F. C. Takahashi, and F. Campelo, "Robust multiobjective optimization using regression models and linear subproblems," in *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 2017, pp. 569–576.
- [16] M. Nasir, A. K. Mondal, S. Sengupta, S. Das, and A. Abraham, "An improved multiobjective evolutionary algorithm based on decomposition with fuzzy dominance," in *Evolutionary Computation (CEC), 2011 IEEE Congress on*. IEEE, 2011, pp. 765–772.
- [17] F. Campelo and C. Aranha, "Moeadr: Component-wise moea/d implementation," URL <https://cran.R-project.org/package=MOEADr>, 2018, r package version 1.2.0.
- [18] F. Campelo, L. Batista, and C. Aranha, "The MOEADr package: A component-based framework for multiobjective evolutionary algorithms based on decomposition," *Journal of Statistical Software*, 2020, in press. Available from: <https://arxiv.org/abs/1807.06731>.
- [19] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," in *Evolutionary multiobjective optimization*. Springer, 2005, pp. 105–145.
- [20] Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, W. Liu, and S. Tiwari, "Multiobjective optimization test instances for the cec 2009 special session and competition," *University of Essex, Colchester, UK and Nanyang technological University, Singapore, special session on performance assessment of multi-objective optimization algorithms, technical report*, vol. 264, 2008.
- [21] J. Bossek, "smoof: Single- and multi-objective optimization test functions," *The R Journal*, 2017. [Online]. Available: <https://journal.r-project.org/archive/2017/RJ-2017-004/index.html>
- [22] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 5, pp. 477–506, 2006.
- [23] H. Li, K. Deb, Q. Zhang, P. Suganthan, and L. Chen, "Comparison between moea/d and nsga-iii on a set of novel many and multi-objective benchmark problems with challenging difficulties," *Swarm and Evolutionary Computation*, vol. 46, pp. 104–117, 2019.
- [24] R. Tanabe, H. Ishibuchi, and A. Oyama, "Benchmarking multi-and many-objective evolutionary algorithms under two optimization scenarios," *IEEE Access*, vol. 5, pp. 19 597–19 619, 2017.
- [25] A. Radulescu, M. López-Ibáñez, and T. Stützle, "Automatically improving the anytime behaviour of multiobjective evolutionary algorithms," in *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, 2013, pp. 825–840.