Bet-and-Run Strategy and MOEA/D An Alternative Resource Management

First Author[1][0000−1111−2222−3333], Second Author[2,3][1111−2222−3333−4444], and Third Author[3][2222−−3333−4444−5555]

[1] Princeton University, Princeton NJ 08544, USA
[2] Springer Heidelberg, Tiergartenstr. 17, 69121 Heidelberg, Germany
lncs@springer.com
http://www.springer.com/gp/computer-science/lncs
[3] ABC Institute, Rupert-Karls-University Heidelberg, Heidelberg, Germany
{abc,lncs}@uni-heidelberg.de

Abstract. Bet-and-run strategies have been shown, both experimentally and theoretically, to be beneficial on single objective problems. Given this success and the fact that they do not take any problem knowledge into account, here we propose an integration of bet-and-run strategy into the multi-objective evolutionary algorithm based on Decomposition framework, (MOEA/D). We conduct a systematic analysis that compares two variations of MOEA/D (the classical MOEA/D and MOEA/D-DE) with and without bet-and-run strategy. Each MOEA is evaluates under 7-DTLZ and 9-WFG benchmark functions and the results indicates that XXX.

1 Introduction

A Multi-objective Optimization Problem (MOP) is box-constrained problems that have m multiple objective functions that must be optimized simultaneously:

$$minf(x) = (f_1(x), f_2(x), ..., f_{n_f}(x)), \text{ subject to } x \text{ in } \Omega, \tag{1}$$

where n_f is the number of objective functions, $x \in \mathbb{R}^{n_v}$, the decision vector, represents a candidate solution with n_v variables, $f : \mathbb{R}^{n_v} \to \mathbb{R}^{n_v}$ is a vector of objective functions and Ω is the feasible decision space. Ω is defined as:

$$\Omega = \{x \text{ in } \mathbb{R}^{n_v} | g_i(x) \leq 0 \ \forall_i \text{ and } h_i(x) = 0 \ \forall_j\}, \tag{2}$$

Objectives often conflict with each other therefore no point in Ω minimizes all the objectives at the same time. Consequently, the goal of MOP solvers is to find the best trade-off that balances the different objectives in an optimal way.

Given two feasible solutions u, v in Ω, u Pareto-dominates v, denoted by $f(u) \prec f(v)$, if and only if $f_k(u) \leq f_k(v), \forall_k \in \{1, ..., n_f\}$ and $f(u) \neq f(x)$. A

solution $x^* \in \Omega$ is considered Pareto-Optimal if there exists no other solution $y \in \Omega$ such that $f(y) \succ f(x^*)$, i.e., if x^* is non-dominated in the feasible decision space. A point is called non-dominated if no other point dominates it. That is, no single solution provides a better trade-off in all objectives.

The set of all Pareto-optimal solutions is known as the Pareto-Optimal set (PS), while the image of this set is referred to as the Pareto-optimal front (PF).

$$PS = x^* \in \Omega | \nexists y \in \Omega : f(y) \succ f(x^*), \tag{3}$$

$$PF = f(x^*) | x^* \in PS. \tag{4}$$

Multi-objective evolutionary algorithms (MOEAs) have been accepted as a basic tool for approximating the PF of a MOP in a single run [27].Over the last two decades, many different search techniques were proposed for improving the effectiveness of multi-objective algorithms. Among them, three are the major paradigms: Pareto domination-based approaches [7], [30], the indicator-based approaches [1], [29], and the decomposition-based approaches [17], [24].

MOEA/D [24] represents a class of population-based meta-heuristics for solving Multi Objective Problems [21]. In this search paradigm, the original multi-objective problem is decomposed into simpler, single-objective sub-problems by means of scalarizations. However, it may waste functions evaluations searching in directions that do not present Pareto-optimal solutions [2].

To reduce this computational cost, the allocation of resources may be better distributed among different sub-problems according to their difficulties. One way to address this problem is to adjust the behavior of the algorithm in an on-line manner to suit the problem in question [12], [6], [19], [25], [16], [23]. All algorithmic components can be tunned adaptively and often feedback information is needed for these adaptation strategies. One variation that aimed to improve this drawback is to use a dynamic/generalized resource allocation (D/G-RA) strategy which allocates resources given the difficulties of the sub-problems studied in MOEA/D-DRA by Zhang [25] and in MOEA/D-GRA Zhou et. al [28].

Another approach that could help reducing the amount of functions evaluations is to use restart strategies. That might help if the algorithm is wasting evaluations searching in directions that are not promising, the algorithm is restarted and a new initial set of solutions is sampled. It is well known that the initial set of solutions often influences the quality outcome, but by simpling re-sample the set of solutions might not direct the algorithm to a more promising region of the search space. Therefore one more elaborated strategy that considers different initial set of solutions may positively address the final quality of the results.

One strategy that consideres different initial set of solutions is the Bet-and-Run strategy [9]. It extends two typical uses for an algorithm with budget t are to use all of the budget for a single run of the algorithm or to make a number of k runs of the algorithm and run each with t/k. In this strategy, k independent runs are all executed with a fraction of the total budget and the best among them continues with the remaining of the total budget.

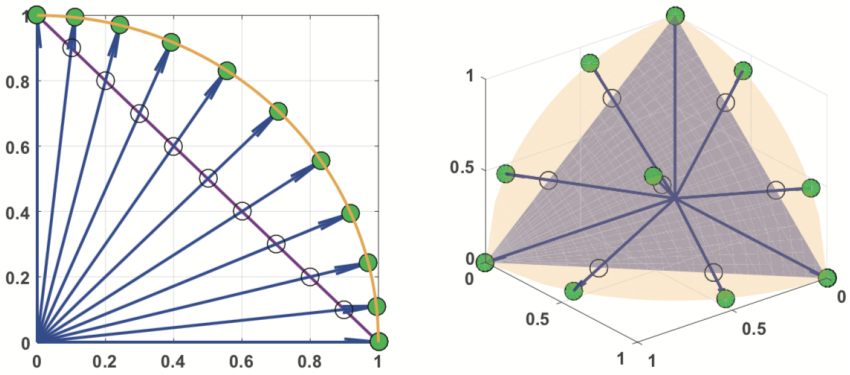

Fig. 1. Decomposition - 2 and 3 objectives - Figure from [5].

From a theoretical point of view, the way that this strategy relates with initialization can be beneficial even on easy functions and be an effective countermeasure against problems with promising but deceptive regions. However, this benefit come with a cost which is the introduction of two additional control parameters, that are non-trivial to be chosen [18].

The main contributions of this paper can be summarized as follows:

- A careful examination of the proper settings of the hyper-parameters of the Bet-and-Run strategy when combined with the two variants (MOEA/D [24] and MOEA/D-DE [17]) of MOEA/D with Bet-and-Run strategy which realize the best performance of the MOEAs on the 10 CEC2009 competition functions [26].
- Proposition an alternative way of choosing the MOEA/D variants hyper-parameters by randomly selecting them.
- An extensive analysis of the performance of using the hyper-volume indicator [31], the epsilon indicator [32], and the IGD indicator [32] on the 7-DTLZ [8] and the 9-WFG [13].

2 MOEA/D

.

Here, MOEA/D [24] is briefly explained and its procedure is shown in algorithm 1. MOEA/D decomposes an MOP with n_f objectives, as defined in equation 1, into N sub-problems using a set of uniformly distributed weight vectors. To define the sub-problem a weight vector is used as a decomposition strategy to define the sub-problems. All these N sub-problems are expected to represent a good approximation to the PF, as it is shown in Figure 1.

MOEA/D solves these subproblems simultaneously by evolving a set of solutions in a single run by using a aggregation function. This function allied with

the weight vectors need to guarantee that a optimal solution to a sub-problem is Pareto-optimal for the MOP and also need to guarantee that the solutions are well distributed in the space of objectives. Therefore, the set of solutions may provide a fair approximation of the Pareto Front.

The relationship between the set of solutions and the sub-problems is formulated as for every sub-problem a unique solution is associated to it, thus the size of the set of solutions is equal to the number of sub-problems. At each interaction, one new candidate solution is generated by applying a sequence of variation operators to the existing solutions. This new solution is then compared with the old one, and the best solution after is maintained as the incumbent solution given the procedure defined by the update strategy.

To compare the solutions or to generate a new one, the algorithm utilizes a neighborhood that defines the limits the exchange of information of a sub-problem between candidates solutions. This neighborhood is defined based on the distance among their weights.

Algorithm 1: General procedure of MOEA/D framework

Input: Objective functions **f**; constraint functions **g**; input parameters
t$\rightarrow$ 0
run $\rightarrow$ TRUE
Initialize the solution set $X^{(t)} = x^1, ..., x^{n_f}$ by random sampling from Ω
Generate weights Λ
while *run* **do**
  Define or update neighborhood $B^i = 1_i, ..., i_{n_f}$
  Copy incumbent solution set $X'^{(t)}$ into $X'^{(t)}$
  for *each variation operator* $v \in V$ **do**
    $X^{(t)} \leftarrow v(X'^{(t)})$
  end
  Evaluate solutions in $X^{(t)}$ and $X'^{(t)}$
  Define next population $X^{(t+1)}$ Update *run* flag t$\rightarrow$ t + 1
end
return $X^{(t)}$, f($X^{(t)}$)

3 Bet-and-Run

3.1 Restart Strategy

Restart Strategies are a mechanism helps the algorithm to explore more in the solution area [22]. For instance, stochastic algorithms and randomized search heuristics may encounter some stagnation before finding a high quality solution. One way to overcome such stagnation is to introduce a restart strategy, since it forcibly changes the search points by restoring the algorithm to its beginning [15]. Also, Restart Strategy might be used to avoid heavy-tailed running

time distributions [11], because if a execution of an algorithm does not conclude within a pre-determined limit or if the solution quality is unsatisfactory, the algorithm is restarted [18]. Finally, it may be considered as an additional speed-up [10].

3.2 Bet-and-Run framework

Fischetti and Monaci [9] investigated the Bet-and-Run framework. They defined it as a number of short runs with randomized initial conditions (the bet-phase) and then bet on the most promising run(the run-phase) and bring it to completion. In their work, they studied the following Bet-and-Run framework:

Phase 1 performs k runs of the algorithm for some short time limit t_1 with $t_1 \leq t/k$.
Phase 2 uses remaining time $t_2 = t - k * t_1$ to continue *only the best run* from the first phase until time out.

In 2017, Lissovoi and Sudholt [18] analyzed this framework theoretically. They investigate it in the context of single objective problems and found that new initializations can have a small beneficial effect even on very easy functions, that this restart strategy might be an effective countermeasure when problems with promising and deceptive regions are encountered.

To the best of our knowledge, the Bet-and-Run framework was only applied with evolutionary algorithms in the context of single objective problems.

4 Experiment Design

4.1 MOEA/D and Bet-and-Run

In this work, the integrate of both frameworks, the MOEA/D the Bet-and-Run, is analized. First, the implementation of MOEA/D is discussed. Then the implementation of the Bet-and-Run followed by the discussion of how to integratate them.

MOEA/D In this paper, two different MOEA/D classical variations found in the literature were studied. They are the original MOEA/D [24] and MOEA/D-DE [17].

The first modification was to change the parameter control H of the simplex-lattice design (SLD) that is used to generate the weight vectors W. For the 2-objective problem benchmark functions it was set as *199*, while for the 3-objective problem benchmark functions, *19*. Those vales for the H parameter were chosen so that the number of sub-problems and the size of incumbent solutions are equal to *200*, following default settings as in the works of Brockhoff et. al [4] and Tanabe et. al [20]. The other modification was to use an external solution archive, that stores the best solutions found during the search process.

Here, the integration of Generic Resource Allocation - On-line Resource Allocation (GRA-ONRA), proposed in the context of MOEA/D-GRA by Zhou et. al [28] was also considered. The resource distribution with GRA-ONRA is allocated using an adaptive strategy aiming to adjust the behavior of an algorithm in on-line manner to suit the problem in question. Although, other strategies were proposed in the work of Zhou et. al, GRA-ONRA was the one that perfomed better among all strategies proposed. This strategy distributes resources in an execution of MOEA/D given different amounts of resources to different sub-problems, following the assumption that some sub-problems can be more difficult to approximate that others.

Bet-and-Run In this work, the Bet-and-Run framework implemented follows the structured showed by Fischetti and Monaci [9]. One adjustment was made in this work that is to better fit Bet-and-Run to the context of MOP and MOEA/D which is defining the budget as the number of interactions, instead of using time as the budget as in the work of Fischetti and Monaci.

5 Experimental Setting

5.1 Bet-and-Run hyper-parameters study

For automatically fine tuning the Bet-and-Run hyper-parameters, we employ the Iterated Racing Procedure (available in the irace R package) [3]. Ten unconstrained test problems from the CEC2009 competition, UF benchmarck functions, [26] are used, with dimensions ranging from 20 to 60. Dimensions 30, 40 and 50 were reserved for testing while all the others were used for training effort. To evaluate a candidate returned by a configuration we used the The hyper-volume (HV) indicator [31].

Before calculating the HV value, the objective function vector $f(x)$ of each $x \in DS$ was scaled between 0 and 1, as suggested in [14]. Therefore the reference point used, based on the work of Ishibuchi et. at was defined given the value H:

- For 2-objectives: $H = 199$, which leads to a reference point of $(1 + 1/H, 1 + 1/H) = (1.005, 1.005)$, meaning $n_f = 200$.
- For 3-objectives: $H = 99$, which leads to a reference point of $(1 + 1/H, 1 + 1/H) = (1.010, 1.010)$, meaning $n_f = 210$.

The tuning budget was set to 20,000 runs for this procedure, while the number of interactions was defined to *300*. The possible configurations are composed from the following choices:

- The number of k runs in Phase 1 perform: in the interval of [2,20].
- The number of interactions for each of the runs in Phase 1: in the interval of [2,10].

5.2 results

6 Comparison Study

The 7-DTLZ [8] and the 9-WFG [13] test problems were used to perform the analysis. For the first the number of objectives used was two and three objectives. According to [8], for the DTLZ problems, the number of position variables k was set to $k = 5$ for the DTLZ1 problem, $k = 7$ for the DTLZ2 problem and $k = 10$ for the other DTLZ problems, where the number of variables $D = n_f + k - 1$. While for the WFG problems the number of position variables D was set to $k = 2(n_f) - 1$, the number of distance variables L was set to $L = 20$ and the number of dimensions is given by $D = k + L$.

The termination criteria was set to the maximum of iterations, with value of *300* interactions, which leads to a number of functions evaluations of *60200* in the case of the 2-objective problems, while for the 3-objective problems, the number of evaluations lead was *63210*.

The hyper-volume indicator [31], the epsilon indicator [32], and the Inverted Generational Distance (IGD) [32] were used for evaluating the quality of the PF found by the MOEAs here studied.set of obtained non-dominated solutions DS

The hyper-volume (HV) indicator [31] was used for evaluating following the same principles and structure as in for the fine tune study.

The epsilon-indicator [32] was used to evaluate the quality of two PF found. This indicator is a binary indicator that compares two Pareto Fronts and may indicate which Front is better and by how much. The first Pareto Front used is the one of the MOEAs while the second is always the *control* Pareto Front given by MOEA/D or MOEA/D-DE. This second Pareto Front is called control since its only purpose is to define a fair reference front for every bet.

Two groups of combinations of algorithms were tested. The first is the MOEA/D group and consists of the following MOEAs: *MOEA/D*, *MOEA/D with Bet-and-Run strategy*, *MOEA/D with GRA-ONRA*, *MOEA/D with Bet-and-Run strategy and GRA-ONRA*.

The other group is the MOEA/D-DE group and consists of: *MOEA/D-DE*, *MOEA/D-DE with Bet-and-Run strategy*, *MOEA/D-DE with GRA-ONRA*, *MOEA/D-DE with Bet-and-Run strategy and GRA-ONRA*.

GRA-ONRA The only extra parameter added by GRA-ONRA is the, DT. Here, it is set to 20, in accord to the findings in [28].

6.1 results

For each combinations above, the search was repeated 30 times, and the Kruskal-Wallis XXX box-plot XXXX.

7 Conclusion

References

1. Beume, N., Naujoks, B., Emmerich, M.: Sms-emoa: Multiobjective selection based on dominated hypervolume. European Journal of Operational Research **181**(3), 1653–1669 (2007)
2. Bezerra, L.C., López-Ibáñez, M., Stützle, T.: Comparing decomposition-based and automatically component-wise designed multi-objective evolutionary algorithms. In: International Conference on Evolutionary Multi-Criterion Optimization. pp. 396–410. Springer (2015)
3. Bezerra, L.C., López-Ibánez, M., Stützle, T.: Automatic component-wise design of multiobjective evolutionary algorithms. IEEE Transactions on Evolutionary Computation **20**(3), 403–417 (2016)
4. Brockhoff, D., Tran, T.D., Hansen, N.: Benchmarking numerical multiobjective optimizers revisited. In: Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation. pp. 639–646. ACM (2015)
5. Chugh, T.: Handling expensive multiobjective optimization problems with evolutionary algorithms. Jyväskylä studies in computing 263. (2017)
6. De Jong, K.: Parameter setting in eas: a 30 year perspective. In: Parameter setting in evolutionary algorithms, pp. 1–18. Springer (2007)
7. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: Nsga-ii. IEEE transactions on evolutionary computation **6**(2), 182–197 (2002)
8. Deb, K., Thiele, L., Laumanns, M., Zitzler, E.: Scalable test problems for evolutionary multiobjective optimization. In: Evolutionary multiobjective optimization, pp. 105–145. Springer (2005)
9. Fischetti, M., Monaci, M.: Exploiting erraticism in search. Operations Research **62**(1), 114–122 (2014)
10. Friedrich, T., Kötzing, T., Wagner, M.: A generic bet-and-run strategy for speeding up stochastic local search. In: AAAI. pp. 801–807 (2017)
11. Gomes, C.P., Selman, B., Crato, N., Kautz, H.: Heavy-tailed phenomena in satisfiability and constraint satisfaction problems. Journal of automated reasoning **24**(1-2), 67–100 (2000)
12. Hinterding, R., Michalewicz, Z., Eiben, A.E.: Adaptation in evolutionary computation: A survey. In: Evolutionary Computation, 1997., IEEE International Conference on. pp. 65–69. IEEE (1997)
13. Huband, S., Hingston, P., Barone, L., While, L.: A review of multiobjective test problems and a scalable test problem toolkit. IEEE Transactions on Evolutionary Computation **10**(5), 477–506 (2006)
14. Ishibuchi, H., Imada, R., Setoguchi, Y., Nojima, Y.: How to specify a reference point in hypervolume calculation for fair performance comparison. Evolutionary computation pp. 1–29 (2018)
15. Kanahara, K., Katayama, K., Okano, T., Kulla, E., Oda, T., Nishihara, N.: A restart diversification strategy for iterated local search to maximum clique problem. In: Conference on Complex, Intelligent, and Software Intensive Systems. pp. 670–680. Springer (2018)
16. Kramer, O.: Evolutionary self-adaptation: a survey of operators and strategy parameters. Evolutionary Intelligence **3**(2), 51–65 (2010)

17. Li, H., Zhang, Q.: Multiobjective optimization problems with complicated pareto sets, moea/d and nsga-ii. IEEE Transactions on evolutionary computation **13**(2), 284–302 (2009)
18. Lissovoi, A., Sudholt, D., Wagner, M., Zarges, C.: Theoretical results on bet-and-run as an initialisation strategy. In: Proceedings of the Genetic and Evolutionary Computation Conference. pp. 857–864. ACM (2017)
19. Meyer-Nieberg, S., Beyer, H.G.: Self-adaptation in evolutionary algorithms. In: Parameter setting in evolutionary algorithms, pp. 47–75. Springer (2007)
20. Tanabe, R., Ishibuchi, H.: An analysis of control parameters of moea/d under two different optimization scenarios. Applied Soft Computing **70**, 22–40 (2018)
21. Trivedi, A., Srinivasan, D., Sanyal, K., Ghosh, A.: A survey of multiobjective evolutionary algorithms based on decomposition. IEEE Transactions on Evolutionary Computation **21**(3), 440–462 (2017)
22. Yu, V., Iswari, T., Normasari, N., Asih, A., Ting, H.: Simulated annealing with restart strategy for the blood pickup routing problem. In: IOP Conference Series: Materials Science and Engineering. vol. 337, p. 012007. IOP Publishing (2018)
23. Zhang, J., Chen, W.N., Zhan, Z.H., Yu, W.J., Li, Y.L., Chen, N., Zhou, Q.: A survey on algorithm adaptation in evolutionary computation. Frontiers of Electrical and Electronic Engineering **7**(1), 16–31 (2012)
24. Zhang, Q., Li, H.: Moea/d: A multiobjective evolutionary algorithm based on decomposition. IEEE Transactions on evolutionary computation **11**(6), 712–731 (2007)
25. Zhang, Q., Liu, W., Li, H.: The performance of a new version of moea/d on cec09 unconstrained mop test instances. In: Evolutionary Computation, 2009. CEC'09. IEEE Congress on. pp. 203–208. IEEE (2009)
26. Zhang, Q., Zhou, A., Zhao, S., Suganthan, P.N., Liu, W., Tiwari, S.: 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 **264** (2008)
27. Zhou, A., Qu, B.Y., Li, H., Zhao, S.Z., Suganthan, P.N., Zhang, Q.: Multiobjective evolutionary algorithms: A survey of the state of the art. Swarm and Evolutionary Computation **1**(1), 32–49 (2011)
28. Zhou, A., Zhang, Q.: Are all the subproblems equally important? resource allocation in decomposition-based multiobjective evolutionary algorithms. IEEE Transactions on Evolutionary Computation **20**(1), 52–64 (2016)
29. Zitzler, E., Künzli, S.: Indicator-based selection in multiobjective search. In: International Conference on Parallel Problem Solving from Nature. pp. 832–842. Springer (2004)
30. Zitzler, E., Laumanns, M., Thiele, L.: Spea2: Improving the strength pareto evolutionary algorithm. TIK-report **103** (2001)
31. Zitzler, E., Thiele, L.: Multiobjective optimization using evolutionary algorithms—a comparative case study. In: International conference on parallel problem solving from nature. pp. 292–301. Springer (1998)
32. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., Da Fonseca, V.G.: Performance assessment of multiobjective optimizers: An analysis and review. IEEE Transactions on evolutionary computation **7**(2), 117–132 (2003)