

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

First Author¹[0000–1111–2222–3333], Second Author^{2,3}[1111–2222–3333–4444], and
Third Author³[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 and are not tailored to the optimization algorithms, here it is proposed an integration a bet-and-run strategy into the Multiobjective Evolutionary Algorithm based on Decomposition framework, (MOEA/D).

MOEA/D represent a class of population-based metaheuristics for the solution of multicriteria optimizarion problems. It decomposes a multi-objective optimization problem into a set of scalar objective subproblems and solve this set in a collaborative way.

1 Introduction

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

$$\min f(x) = (f_1(x), f_2(x), \dots, f_{n_f}(x)), \text{ subject to } x \text{ in } \Omega, \quad (1)$$

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, $f : \mathbb{R}^{n_v} \rightarrow \mathbb{R}^{n_f}$ is a vector of objective functions and Ω is the feasible decision space, such that:

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

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 $f(u) \succ f(v)$, if and only if $f_k(u) \leq f_k(v), \forall_k \in \{1, \dots, n_f\}$ and $f(u) \neq f(v)$. A

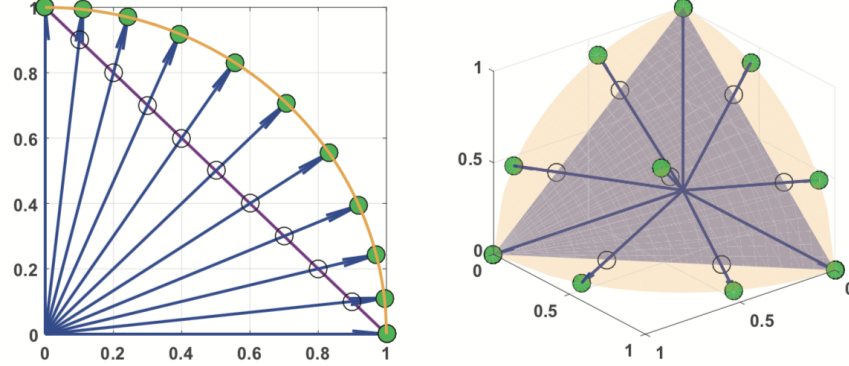


Fig. 1. Decomposition - 2 and 3 objectives - Figure from [2]

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).

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

$$PF = f(x^*) \mid x^* \in PS. \quad (4)$$

2 MOEA/D

Here, MOEA/D [14] is briefly explained. MOEA/D represents a class of population-based meta-heuristics for solving Multi Objective Problems (MOPs). It is based on decomposition, which is one kind of scalarizing function, where one multi-objective problem becomes various single-objective sub-problem. That is, 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 2.

MOEA/D solves these subproblems simultaneously by evolving a set of solutions in a single run by using an 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 \mathbf{f} ; constraint functions \mathbf{g} ; input parameters

```

1  $t \leftarrow 0$ 
2  $run \rightarrow \text{TRUE}$ 
3 Initialize the solution set  $X^{(t)} = x^1, \dots, x^{n_f}$  by random sampling from  $\Omega$ 
  Generate weights  $\lambda$ 
4 while  $run$  do
5   Define or update neighborhood  $B^i = 1_i, \dots, i_{n_f}$ 
6   Copy incumbent solution set  $X'^{(t)}$  into  $X^{(t)}$ 
7   for each variation operator  $v \in V$  do
8      $X^{(t)} \leftarrow v(X^{(t)})$ 
9   end
10  Evaluate solutions in  $X^{(t)}$  and  $X'^{(t)}$ 
11  Define next population  $X^{(t+1)}$  Update  $run$  flag  $t \leftarrow t + 1$ 
12 end
13 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 [13]. 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 [9]. Also, Restart Strategy might be used to avoid heavy-tailed running time distributions [6], 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 [11]. Finally, it may be considered as an additional speed-up [5].

3.2 Bet-and-Run framework

Fischetti and Monaci [4] 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 bet-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 [11] 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 combinations found in the literature were studied. These combinations are the original MOEA/D [14] and MOEA/D-DE [10].

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 [1] and Tanabe et. al [12]. The other modification was to use an external solution archive, that stores the best solutions found during the search process.

Here, the integration of On-line Resource Allocation (ONRA), proposed in the context of MOEA/D by [15] was also considered. The resource distribution

when using 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, ONRA was the one that performed better among all strategies proposed. The ONRA strategy is concerned with the distribution of resources in an execution of MOEA/D. Different amounts of resources are considered to different sub-problems, following the assumption that some sub-problems can be more difficult to approximate than others.

Bet-and-Run In this work, the Bet-and-run framework implemented follows the results found by Friedrich et. al [5]. They studied different combinations strategies that are diverse on the amount of resources assigned for phase 1 and 2.

The best overall strategy found is the one that uses 40% of the total budget available on short runs (phase 1) and then run the most prominent one (phase 2) with the remaining 60% of the budget found. One adjustment was made to better fit 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 Friedrich et. al used.

The ϵ -indicator[17] was used to evaluate the quality the bet-instances and to choose among them, the most prominent one. It is a binary indicator that compares 2 Pareto Fronts and may indicate which Front is better and by how much. The first Pareto Front used is the one from the bet-instances while the second is the *control* Pareto Front. This second Pareto Front is called control since its only purpose is to define a fair reference front for every bet.

For this experiments, the control Pareto Front is generate every time the Bet-and-Run+MOEA/D is run. It is based on the same algorithm configuration as the Bet-and-Run+MOEA/D and is executed for only one interaction. After all bet-instances are evaluated, the one with the lowest value is chosen as the most prominent one.

ONRA The only extra parameter added by ONRA is the, DT . Here, it is set to 20, in accord to the findings in [15].

4.2 Experimental Setting

The DTLZ [3] and the WFG [7], test problems were used to perform the analysis. For the first the number of objectives used was two and three objectives. According to [3], 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 $D = k + L$.

Here 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 (HV) indicator [16] was used for evaluating the quality of a set of obtained non-dominated solutions DS . 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 [8]. 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 $(1 + 1/H, 1 + 1/H) = (1.005, 1.005)$
- For 3-objectives: $H = 99$, which leads to a reference point of $(1 + 1/H, 1 + 1/H) = (1.010, 1.010)$

The following combinations of algorithms (MOEA/D-DE or MOEA/D) were tested: *MOEA/D-DE*, *MOEA/D-DE with Bet-and-Run strategy*, *MOEA/D-DE with ONRA*, *MOEA/D-DE with Bet-and-Run strategy and ONRA*, *MOEA/D*, *MOEA/D with Bet-and-Run strategy*, *MOEA/D with ONRA*, *MOEA/D with Bet-and-Run strategy and ONRA*.

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

5 Results

Analysis are done with

6 Conclusion

References

1. 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)
2. Chugh, T.: Handling expensive multiobjective optimization problems with evolutionary algorithms. Jyväskylä studies in computing 263. (2017)
3. Deb, K., Thiele, L., Laumanns, M., Zitzler, E.: Scalable test problems for evolutionary multiobjective optimization. In: Evolutionary multiobjective optimization, pp. 105–145. Springer (2005)
4. Fischetti, M., Monaci, M.: Exploiting erraticism in search. Operations Research **62**(1), 114–122 (2014)
5. Friedrich, T., Kötzling, T., Wagner, M.: A generic bet-and-run strategy for speeding up stochastic local search. In: AAAI. pp. 801–807 (2017)
6. 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)

7. 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)
8. 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)
9. 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)
10. 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)
11. 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)
12. Tanabe, R., Ishibuchi, H.: An analysis of control parameters of moea/d under two different optimization scenarios. *Applied Soft Computing* **70**, 22–40 (2018)
13. 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)
14. Zhang, Q., Li, H.: Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on evolutionary computation* **11**(6), 712–731 (2007)
15. 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)
16. 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)
17. 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)