Testing of the Multi-objective Alliance Algorithm on Benchmark Functions

Valerio Lattarulo
University of Cambridge
Department of Engineering
Engineering Design Centre
Cambridge, Trumpington
Street, CB2 1PZ, UK
vl261@cam.ac.uk

Geoffrey T. Parks
University of Cambridge
Department of Engineering
Engineering Design Centre
Cambridge, Trumpington
Street, CB2 1PZ, UK
gtp10@cam.ac.uk

ABSTRACT

A new version of the Multi-objective Alliance Algorithm (MOAA) is described. The MOAA's performance is compared with that of NSGA-II using the epsilon and hypervolume indicators to evaluate the results. The benchmark functions chosen for the comparison are from the ZDT and DTLZ families and the main classical multi-objective (MO) problems. The results show that the new MOAA version is able to outperform NSGA-II on almost all the problems.

Categories and Subject Descriptors

G.1.6 [**Mathematics of Computing**]: Optimization—*Unconstrained Optimization*; G.4 [**Mathematics of Computing**]: Mathematical Software—*Algorithm design and analysis*

Keywords

MOAA; Multi-objective Optimization; ZDT; DTLZ

1. INTRODUCTION

Real-world optimization problems are generally characterized by several objectives in conflict. Such problems are often tackled using metaheuristic approaches.

The Alliance Algorithm (AA) is a relatively new single-objective optimization algorithm that has been applied successfully to benchmark tests [7],[9] and real-world problems [2],[12]. These promising results motivated the development of a MO variant [10] the performance of which has been compared with well-known algorithms. Since then, a mixed-integer version of the MOAA with hybrid components has been developed. This was able to outperform a hybrid version of NSGA-II on a satellite constellation refueling optimization problem [13]. The knowledge acquired in solving benchmark and complex real-world problems led to the development of a new version of the MOAA, which is presented in this paper. This version has already been used to solve several real-world problems, one of which – the optimization of a baseline two-dimensional airfoil for supersonic flight conditions – will be presented at GECCO 2013 [11].

In this paper, the new MOAA version is tested on benchmark problems with two and three objectives and is found to outperform NSGA-II on almost all the problems.

The remainder of this paper is structured as follows: Sect. 2 describes how the new MOAA version works; Sect. 3 introduces the benchmark problems, the indicators and the statistical tests used for the evaluation of the algorithms, reports on the MOAA's performance on these problems, comparing it with that of NSGA-II, and discusses the results; Sect. 4 concludes the paper and suggests possible future work.

2. ALGORITHM

The MOAA is a metaheuristic optimization algorithm inspired by the metaphorical idea of a number of tribes struggling to conquer an environment that offers resources that enable them to survive. The tribes are characterized by two features: the skills and resources necessary for survival. Tribes try to improve skills by forming alliances, which are also characterized by the skills and resources needed, but these now depend on the tribes within the alliance. The two main search elements of the algorithm are the formation of alliances and the creation of new tribes. One MOAA cycle ends when the strongest possible alliances of existing tribes have been created. The algorithm then begins a new cycle starting with new tribes whose creation is influenced by the previous strongest alliances.

A tribe t is a tuple (x_t, s_t, r_t, a_t) composed of: a point in the solution space x_t; a set of skills dependent on the values of the N_S objective functions evaluated at x_t; a set of resource demands dependent on the values of the N_R constraint functions; an alliance vector a_t containing the IDs of the tribes allied to tribe t.

An alliance is a mutually disjoint partition of tribes. Each alliance a forms a new point x_a in the solution space defined by the tribes in the alliance. The sets of skills s_a and resource demands r_a of the alliance consist of the objective and constraint functions S and R evaluated at x_a.

2.1 Algorithm Steps

The procedure followed by the MOAA can be divided into several steps. For reasons of space, only a general description of the steps without equations is provided. A detailed

GECCO'13 Companion, July 6–10, 2013, Amsterdam, The Netherlands.

description of these steps can be found in [11]. A general definition of the framework is provided in [8], a copy of which is available from the first author on request.

2.1.1 Solution Generation

In the MOAA's first cycle the tribes (solutions) are chosen randomly (with a uniform distribution). In subsequent cycles, one of the Pareto-optimal (PO) solutions found is chosen randomly and normalized; with probability P_1, a new solution is simply a copy of this PO solution; otherwise, with probability P_2, the variables are modified, on an individual basis, with a normal distribution with standard deviation σ around the chosen point, or, with probability $1 - P_2$, they assume the same values as those of the chosen point. This cycle is repeated until N tribes have been generated.

An important feature of this new version of the MOAA is the adaptive nature of σ: this parameter adaptively decreases in order to produce high diversity at the beginning of the optimization and low diversity at the end. This mechanism enhances the initial exploration of the solution space and the final convergence of the solutions already found.

2.1.2 Verification

In this phase an alliance/tribe (A/T) is chosen randomly and given the chance to forge an alliance by being given a token. Meanwhile all the other A/Ts wait their turn. The A/T t with the token chooses another tribe to become an ally, thus forming a new alliance (a point in solution space x_a). When an alliance is created, x_a is made up of components drawn from the tribes within the alliance: given an alliance of N_a tribes, each component of x_a has a $1/N_a$ probability of being equal to the corresponding component of any tribe in the alliance. There is an additional probability P_3 that the component is then modified, applying a variation. This is essentially a uniform recombination with a variation applied randomly to some of the variables. This operation is repeated until all the components of x_a are defined.

The standard deviation for the variation depends on the distance between the highest and lowest values of the corresponding variable among the tribes within the alliance, the variation for an alliance of tribes that are close together is small (local search) and for tribes that are far apart it is large (global search). Generally at the start of an optimization the tribes within an alliance are far apart and then they start to come closer together. This behavior can be viewed as an initial global search followed by progressively more localized search.

The new alliance will only be confirmed if at least one skill in s_a of x_a is better than one skill in s_{t_1} of the solution representing the A/T with the token x_{t_1} and one skill in s_{t_2} of the tribe chosen to become an ally.

The resource function $R(x)$ plays no role here because the benchmark problems tackled in this particular study are unconstrained.

2.1.3 Alliance and Data Structure Update

There are two possible outcomes from the Verification Phase: the chosen tribe joins the entity with the token forming a new alliance, or the tribe does not join and the new alliance is not confirmed. Next there is an update of the data structures necessary for the low level system to function, such as the necessity to provide a unique ID to the created alliances. The cycle termination conditions are also checked. The cycle finishes when each A/T has tried to form a new alliance with every other tribe and remains unchanged (because there is no advantage in changing). If this condition is not met, the algorithm returns to the Token Phase.

2.1.4 Selection of the Strongest Alliances and Termination

At the end of the interactions between tribes, many alliances will have been formed but only the strongest A/Ts will conquer the environment. Therefore the A/Ts selected are the non-dominated points in objective space. These correspond to the best solutions to the problem found thus far. They can be used as the input to another MOAA cycle or, if the algorithm has ended, they represent the final results.

There is a limit n to the number of best solutions saved in the archive of PO solutions. If the number of non-dominated solutions exceeds this, then all the solutions with at least one neighbor within a neighborhood distance d (in objective space) are eliminated. The initial value of d is 0 and then changes adaptively. This formulation takes into account the reality that the rates of convergence have to depend on the number of function evaluations that can be afforded: slower convergence is possible if many function evaluations are possible; faster convergence is necessary with few function evaluations. Moreover it also depends on the actual ranges of the Pareto front: larger ranges require faster rates in order that the number of accepted solutions in the archive does not explode; conversely, smaller ranges require slower rates. In this way a balance around n is struck between the new PO solutions found by the algorithm every cycle and the solutions removed from the archive of PO solutions.

The MOAA is terminated when a specified limit E_{tot} on the number of solution evaluations is reached. The output of the algorithm is then the best solutions and the Pareto front found.

2.1.5 Extended Archive

Another innovation in this new MOAA version is the use of an Extended Archive. The archive with the PO solutions records only the non-dominated solutions, not considering many solutions that could help maintain diversity among solutions and the convergence of the algorithm. This issue is addressed by allowing other solutions to join the archive in the way described below.

For each dimension in objective space, the PO solutions and the solutions found in the cycle are sorted by increasing objective values.

The distance (the difference in objective values for the relevant objective) between consecutive PO solutions is calculated as is the distance between the last PO solution (that with the worst objective value) and the solution from the current cycle with the worst value of that objective, in order to include the edge of the function.

If the distance between two solutions is greater than the mean distance multiplied by a factor d_f, then non-dominated sorting of the solutions from the current cycle within that particular range is done and the non-dominated solutions from this group are added to the archive.

The factor d_f changes adaptively, increasing over time because the gaps between solutions become smaller, reaching similar values: as d_f is increased only large gaps (in comparison with the average) will be taken into consideration.

3. RESULTS

In this section the performance of the new MOAA version is compared with that of NSGA-II [4]. The new MOAA version was also able to outperform the preliminary version [10] on these problems but the results are not presented for reasons of space. The main differences between the new and preliminary versions are the use of adaptive parameters, improved variation operators and the Extended Archive.

Each algorithm was tested 50 times on each problem, with a function evaluation limit for each run of 10,000 for two-objective problems and 30,000 for three-objective problems. The results from each set of 50 runs were used to calculate the mean and standard deviation of each indicator and in the statistical tests.

16 problems with a range of characteristics were selected to test the algorithms. The test problems can be divided into three categories: the ZDT family [14], the DTLZ family [5] and the main classical MO problems [3]. The ZDT family and the main classical problems are bi-objective problems; the DTLZ family consists of three-objective problems. The problems chosen from the ZDT family are ZDT1-ZDT4 and ZDT6; those from the DTLZ family are DTLZ1-DTLZ7; the classical problems are SCH, FON, POL and KUR.

The performance measures chosen to evaluate the algorithms are the *epsilon indicator* [15] and *hypervolume indicator* [6] provided in the PISA package [1]. The statistical test chosen for the evaluation of results is the Kruskal-Wallis test, provided in the PISA package [1]. This outputs p-values which can be interpreted as the probability that the MOAA is superior to NSGA-II. When the null hypothesis (H0) is not rejected, there is no statistically significant difference between the performance of the two algorithms.

The values of the parameters of the MOAA have already been used to solve several real-world problems. Thus, these values are general and have not been overfitted to these problems. Some parameters need further investigation to find better general values and provide appropriate justifications. Some guidelines can be found in [11]. The parameter values used in this study are given in Table 1.

Table 1: MOAA Parameter Values

Parameter	Value	Description
N	8	Number of tribes
P_1	0.5	Probability 1 for creation of tribes
P_2	0.2	Probability 2 for creation of tribes
σ_{init}	0.3	Initial standard deviation
σ_{end}	0.001	Final standard deviation
P_3	$2/V$	Probability for creation of alliances
σ_a	0.1	Std for creation of alliances
N_{tot}	100	Total number of PO solutions
N_f	10	Factor for evaluation neighborhood

Tables 2 and 3 show the means and standard deviations of the epsilon and hypervolume indicators. It can be observed that the new MOAA version obtains better results on 6 out of 9 bi-objective problems, especially on ZDT2, ZDT6, SCH and POL where its results are remarkably better; in the cases where the MOAA results are worse, the performance of NSGA-II is better but not hugely so. For the three-objective problems, the new MOAA version obtains better overall results on all problems with only a few cases of similar performance.

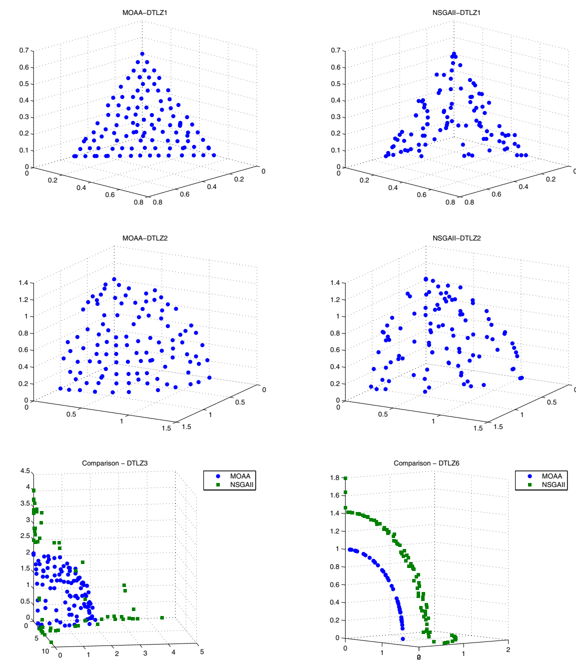

Figure 1: Comparisons of MOAA and NSGA-II Pareto fronts for DTLZ1-2-3-6

There are two types of improvement: on problems like DTLZ1-2-4-5 both algorithms are able to converge to the true Pareto front, but the MOAA is able to spread the solutions better; on problems like DTLZ6 the MOAA is able consistently to converge to (or near to) the true Pareto front, whereas NSGA-II does so inconsistently. These two cases are illustrated in Fig. 1 where the differences between the typical Pareto fronts found by the MOAA and NSGA-II on problems DTLZ1-2-3-6 are shown.

Table 2: Comparison with the epsilon indicator

	MOAA		NSGA-II	
Function	Mean	Std	Mean	Std
SCH	**0.0045**	**0.0007**	0.2430	0.1426
FON	0.0213	0.0131	**0.0130**	**0.0018**
POL	**0.0082**	**0.0013**	0.0748	0.1333
KUR	**0.0111**	**0.0029**	0.1652	0.0203
ZDT1	0.0194	0.0259	**0.0128**	**0.0021**
ZDT2	**0.0111**	**0.0017**	0.1771	0.3603
ZDT3	**0.0459**	**0.0670**	0.0774	0.1045
ZDT4	0.2273	0.1044	**0.1526**	**0.0852**
ZDT6	**0.0032**	**0.0005**	0.0319	0.0022
DTLZ1	**0.0185**	**0.0035**	0.0293	0.0058
DTLZ2	**0.0810**	**0.0116**	0.1311	0.0211
DTLZ3	0.0093	**0.0043**	**0.0077**	0.0059
DTLZ4	**0.0900**	0.0323	0.1178	**0.0159**
DTLZ5	**0.0111**	**0.0017**	0.0148	0.0027
DTLZ6	**0.0036**	**0.0004**	0.1679	0.0629
DTLZ7	**0.0726**	**0.0082**	0.1183	0.0298

Table 4 shows the probability that the MOAA outperforms NSGA-II applying the Kruskal-Wallis test to the results obtained for the epsilon and hypervolume indicators

Table 3: Comparison with hypervolume indicator

	MOAA		NSGA-II	
Function	Mean	Std	Mean	Std
SCH	**0.0006**	**0.0001**	0.2217	0.1437
FON	0.0093	0.0019	**0.0075**	**0.0004**
POL	**0.0031**	**0.0003**	0.0164	0.0253
KUR	**0.0060**	**0.0016**	0.0435	0.0098
ZDT1	0.0068	0.0058	**0.0059**	**0.0003**
ZDT2	**0.0052**	**0.0004**	0.0727	0.1557
ZDT3	**0.0079**	**0.0087**	0.0140	0.0218
ZDT4	0.2118	0.1098	**0.1465**	**0.0958**
ZDT6	**0.0012**	**0.0009**	0.0286	0.0019
DTLZ1	**0.0012**	0.0005	0.0019	**0.0004**
DTLZ2	**0.0486**	**0.0026**	0.0787	0.0085
DTLZ3	**0.0009**	**0.0017**	0.0011	0.0030
DTLZ4	**0.0524**	0.0156	0.0729	**0.0044**
DTLZ5	**0.0042**	0.0008	0.0045	**0.0002**
DTLZ6	**0.0010**	**0.0001**	0.1408	0.0589
DTLZ7	**0.0345**	**0.0016**	0.0510	0.0065

by the algorithms. This test confirms the superiority of the MOAA, showing that it performs better (probability 1 or near to 1) on almost all the problems. The test also confirms that, in most of its worse cases, its performance is still comparable to that of NSGA-II (ZDT3, ZDT4 and DTLZ3). There are many cases where the p-value is 1 for both the epsilon and hypervolume indicators (ZDT2-6, SCH, POL, KUR, ZDT6, DTLZ2-4-6-7); in other cases the p-value is near to 1 (ZDT1, DTLZ1-5). In only one case, the hypervolume of FON, is the p-value 0.

Table 4: Kruskal-Wallis test between the new MOAA version and NSGA-II

Function	eps	hyp	Function	eps	hyp
ZDT1	H0	1	KUR	1	1
ZDT2	1	1	DTLZ1	1	0.99
ZDT3	H0	H0	DTLZ2	1	1
ZDT4	H0	H0	DTLZ3	H0	H0
ZDT6	1	1	DTLZ4	1	1
SCH	1	1	DTLZ5	0.99	0.99
FON	H0	0	DTLZ6	1	1
POL	1	1	DTLZ7	1	1

4. CONCLUSIONS AND FUTURE WORK

This paper has described a new version of the Multi-objective Alliance Algorithm (MOAA). Its performance has been compared with NSGA-II on various benchmark problems, using the epsilon and hypervolume indicators to evaluate the results and the Kruskal-Wallis test for statistical analysis. The results show that the new MOAA version outperforms NSGA-II in almost all cases. This study aims to show that the new MOAA version, already applied to some real-world problems using the same or very similar parameter values, also performs well on benchmark tests.

In future work, it would be interesting to compare this approach with other algorithms that perform well on these benchmark problems or on real-world problems. It would also be of value to develop comprehensive guidelines for the choice of MOAA parameter values for different problems.

5. REFERENCES

[1] S. Bleuler, M. Laumanns, L. Thiele, and E. Zitzler. PISA—A platform and programming language independent interface for search algorithms. In *Evolutionary Multi-Criterion Optimization (EMO 2003)*, 2003.

[2] V. Calderaro, V. Galdi, V. Lattarulo, and P. Siano. A new algorithm for steady state load-shedding strategy. In *12th International Conference on Optimization of Electrical and Electronic Equipment (OPTIM)*, pages 48–53, 2010.

[3] C. A. Coello Coello, B. G. Lamont, and A. D. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer, New York, NY, USA, second edition, 2007.

[4] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6:182–197, 2002.

[5] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable test problems for evolutionary multi-objective optimization. Technical report, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, 2001.

[6] J. Knowles, L. Thiele, and E. Zitzler. A tutorial on the performance assessment of stochastic multiobjective optimizers. Technical report, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, 2006.

[7] V. Lattarulo. Application of an innovative optimization algorithm for the management of energy resources. BSc thesis, University of Salerno, 2009.

[8] V. Lattarulo. Multi-objective alliance algorithm. Technical Report CUED/C-EDC/TR.157, Department of Engineering, University of Cambridge, 2011.

[9] V. Lattarulo. Optimization of biped robot behaviors by 'alliance algorithm'. Master's thesis, University of Hertfordshire, 2011.

[10] V. Lattarulo and G. T. Parks. A preliminary study of a new multi-objective optimization algorithm. In *International Conference on Evolutionary Computation (CEC)*, 2012.

[11] V. Lattarulo, P. Seshadri, and G. T. Parks. Optimization of a supersonic airfoil using the multi-objective alliance algorithm. In *GECCO 2013*, 2013.

[12] V. Lattarulo and S. G. van Dijk. Application of the "Alliance Algorithm" to energy constrained gait optimization. In *15th Annual RoboCup International Symposium*, Istanbul, Turkey, 2011.

[13] V. Lattarulo, J. Zhang, and G. T. Parks. Application of the MOAA to satellite constellation refueling optimization. In *Evolutionary Multi-Criterion Optimization (EMO 2013)*, 2013.

[14] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8:173–195, 2000.

[15] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. Grunert da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, 2003.