Who's better? PESA or NSGA II?

Laura Dioşan, Mihai Oltean
Department of Computer Science, Faculty of Mathematics and Computer Science,
Babes-Bolyai University,
Kogălniceanu 1, Cluj-Napoca, 400084,Romania
{*lauras, moltean*}*@cs.ubbcluj.ro*

Abstract

According to the No Free Lunch (NFL) theorems all black-box algorithms perform equally well when compared over the entire set of optimization problems. An important problem related to NFL is finding a test problem for which a given algorithm is better than another given algorithm. In this paper we propose an evolutionary approach for solving this problem: we will evolve multi-objective test functions for which a given algorithm A is better than another given algorithm B. The evolved functions are represented as binary strings. Several numerical experiments involving PESA and NSGA II are performed. The results show the effectiveness of the proposed approach. Several multi-objective problems for which PESA performs better than NSGA II and several multi-objective test problems for which NSGA II performs better than PESA have been evolved.

1. Introduction

Since the birth of the No Free Lunch (NFL) theorems in 1995 [15, 16], the trends of Evolutionary Computation [7] have not changed at all, although these breakthrough theories should have produced dramatic changes. Most researchers chose to ignore these theories: they developed new algorithms that work better than the old ones on some particular test problems. The researchers have eventually added: "The algorithm X performs better than another algorithm *on the considered test problems*". That is somehow useless since the proposed algorithms cannot be the best on all the considered test problems. Moreover, most of the problems employed for testing algorithms are artificially constructed.

Consider for instance, the field of evolutionary single-criteria optimization where most of the algorithms were tested and compared on some artificially constructed test problems (most of them being known as De'Jong test problems) [6, 17]. These test problems were used for comparison purposes before the birth of the NFL theorems and they are used even today (12 years later after the birth of the NFL theorems). Evolutionary multi-criteria optimization was treated in a similar manner: most of the recent algorithms in this field were tested on several artificially constructed test problems proposed in [2, 4, 8].

Roughly speaking, the NFL theorems state that all the black-box optimization algorithms perform equally well over the entire set of optimization problems. Thus, if an algorithm A is better than another algorithm B on some classes of problems, the algorithm B is better than A on the rest of the problems.

As a consequence of the NFL theories, even a computer program (implementing an Evolutionary Algorithm (EA)) containing programming errors can perform better than some other highly tuned algorithms for some test problems.

Random search (RS) being a black box search / optimization algorithm should perform better than all of the other algorithms for some classes of test problems. Even if this statement is true, there is no presentation in the specialized literature of a test problem for which RS performs better than all the other algorithms (taking into account the NFL restriction concerning the number of distinct solutions visited during the search). However, a problem which is hard for all Evolutionary Algorithms is presented in [5].

Three questions (on how we match problems to algorithms) are of high interest:

- For a given class of problems, which is (are) the algorithm(s) that performs (perform) better than all other algorithms?
- For a given algorithm which is (are) the class(es) of problems for which the algorithm performs best?
- Given two algorithms A and B, which is (are) the class (es) of problems for which A performs better than B?

Answering these questions is not an easy task. All these problems are still open questions and they probably lie in

0-7695-2976-3/07 $25.00 © 2007 IEEE
DOI 10.1109/ISDA.2007.106

the class of the NP-Complete problems. If this assumption is true it means that we do not know if we are able to construct a polynomial algorithm that takes a problem as input and outputs the best optimization algorithm for that problem (and vice versa). Fortunately, we can try to develop a heuristic algorithm able to handle this problem.

In this paper we develop a framework for constructing multi-objective test problems that match a given algorithm. More specific, given two evolutionary algorithms A and B, the question is *What are the multi-objective problems for which A performs better than B (and vice-versa)*. The involved algorithms are PESA [9] and NSGA II [3]. For obtaining such problems we will use an evolutionary approach: the functions matched to a given algorithm are evolved by using the Genetic Algorithms (GAs) [7].

The paper is organized as follows: in Section 2 we briefly review NFL-related research. Section 3 provides some standard preliminaries and notation. A metric for performance is presented in Section 4. The evolutionary model used for discover test problems represented as binary strings and the fitness assignment process are described in Section 5. The algorithms used for comparison are described in Section 6. Several numerical experiments are carried out in Section 7. Finally, Section 8 concludes our paper and summarizes the further work directions.

2. Related work

There are several important proofs of NFL since Wolpert have proposed for the first time the theorems in 1995 (for search [15]) and 1997 (for optimisation [16]).

For instance, in 1995 Radcliffe [12] has reviewed the implications of NFL with a focus on representation issues.

More recently, in 2000, Whitley [14] showed that a general NFL result holds over the set of permutation functions, that is, problems in which every point in the search space has a unique objective function value.

A 'sharpened' NFL result is proved in [13], which shows that the NFL holds over a set of functions F, if and only if F is closed under permutation.

In 2004, Oltean [11] has proposed an evolutionary approach for finding a single-objective function for which Random Search is better than another standard evolutionary algorithm: he has evolved single-objective test functions for which a given algorithm is better than another given algorithm. Two ways for representing the evolved functions were employed: as Genetic Programming (GP) trees and as binary strings. The results of the numerical experiments have shown the effectiveness of his approach: functions for which Random Search performs better than other evolutionary algorithms have been successfully discovered. The curent paper extends the results presented in [11] for the multiobjective problems.

Meanwhile, universally, NFL theorems are discussed in terms of single-objective optimization problems. One exception can be found is the early work of Radcliffe [12], which once notes that the domain of the problems concerned could be multi-objective. More recently (in 2003), Corne and Knowles [1] have confirmed that the classic NFL theorem holds for general multi-objective fitness spaces, and have shown how this follows from a 'single-objective' NFL theorem. They have also shown that, given any particular Pareto Front, an NFL theorem holds for the set of all multi-objective problems which have that Pareto Front.

3. The search space

Using similar notation to that in [16, 1], we have a multi-objective optimisation problem f with k objectives that can be given as a mapping $f : X \to Y$, where X is the search space and Y a set of 'fitnesses'; we will generally assume minimization for all the k criteria. $F = Y^X$ is the space of all problems.

If the mapping f is a multi-objective one, than it can be written as $f = (f^1, f^2, \dots f^k)$ and, in this case, we can write Y as:

$$Y = Z_1 \times Z_2 \times \dots \times Z_k.$$

Without loosing the generality, in our numerical experiments we have assumed that $Z_1 = Z_2 = \dots = Z_k = Z$. In this case, Y becomes Z^k.

The size of F is clearly $|Y|^{|X|} = k \times |Z|^{|X|}$.

Because our aim is to find (by evolution's help) some problems for which an algorithm performs better than another one, we will evolve multi-objective test functions represented as binary strings. We employed this representation for the test functions because in this way we can evolve any function without being constrained to a given set of operators as in the case of functions generated using GP [10].

In this case, our analysis is performed in the finite search space X [14]. The space of possible "cost values", Y^k , is also finite. The restriction to the binary search spaces is not a hard one since all other values can be represented as binary strings. Thus $X = \{0,1\}^n$, $Z = \{0,1\}^m$ and, consequently,

$$Y = \underbrace{\{0,1\}^m \times \{0,1\}^m \times \dots \times \{0,1\}^m}_{k \text{ times}}.$$

As we already mentioned, a multi-objective optimization problem f is represented as a mapping $f : X \to Y$, or as $f : X \to Z^k$. The set $F = Y^X$ denotes the space of all possible problems. The size of F is $|Y|^{|X|} = k \times |Z|^{|X|}$.

In our experiments $n = 16$ and $m = 8$. Thus $|X| = 2^{16} = 65536$ and $|Z| = 2^8 = 512$. The number of optimization problems in this class is $|Y|^{|X|} = k \times (2^8)^{65536}$. Each test problem in this class can be stored in a string of

$$|X| \times k \times 8bits = 65536 \times k \times 8bits.$$

Within this huge search space we will try to find test problems for which a given algorithm A is better than another given algorithm B.

4. Metrics of performance

Many metrics for measuring the convergence of a set of nondominated solutions towards the Pareto front have been proposed. Almost all of these metrics were constructed in order to directly compare two sets of nondominated solutions. There are also approaches which compare a set of nondominated solutions with a set of Pareto optimal solutions if the true Pareto front is known.

In our case, we need the first metric type that compares two sets of nondominated solutions. For this purpose we have chosen the S metric.

The S metric has been introduced by Zitzler in [19] and improved in [18]. It measures how much of the objective space is dominated by a given nondominated set NdS or, with other words, the size of the dominated space.

Let X be the set of decision vectors for a particular problem and $NdS = \{x_1, x_2, \ldots, x_t\} \subseteq X$ a set of t decision vectors. The function $S(NdS)$ gives the volume enclosed by the union of the polytopes $p_1, p_2, \ldots, p_t$, where each p_i is formed by the intersection of the following hyper planes arising out of x_i, along with the axes: for each axis in the objective space there exist a hyper plane perpendicular to the axis and passing through the point $(f_1(x_i), f_2(x_i), \ldots, f_k(x_i))$.

In the two-dimensional case, each p_i represents a rectangle defined by the points $(0, 0)$ and $(f_1(x_i), f_2(x_i))$. An example for two-dimensional case is presented in Figure 1.

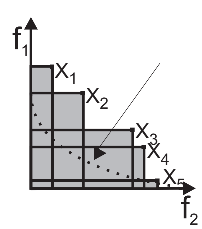

Figure 1. The S metric for the case of two objective functions and 5 decision vectors ($x_1, x_2, \ldots, x_5$) for a minimization problem.

In the three-dimensional case, each polytope p_i represents a parallelepiped defined by the points $(0, 0, 0)$ and $(f_1(x_i), f_2(x_i), f_3(x_i))$.

5. Evolutionary model and the fitness assignment process

5.1. The model

Our aim is to find a multi-objective test function for which a given algorithm A performs better than another given algorithm B. The multi-objective test function that is being searched for will be represented as strings over the $\{0, 1\}$ alphabet as described in section 3.

The algorithm used for evolving these functions is a standard steady state GA that works with a binary encoding of individuals [7]. Each multi-objective test problem in this class can be stored in a string of $65536 \times k \times 8bits$, where k represents the number of objectives.

The most important aspect of this algorithm regards the way in which the fitness of an individual is computed. The quality of the test function encoded in a chromosome is computed as follows: The given algorithms A and B are applied to the test function. These algorithms will try to optimize that test function.

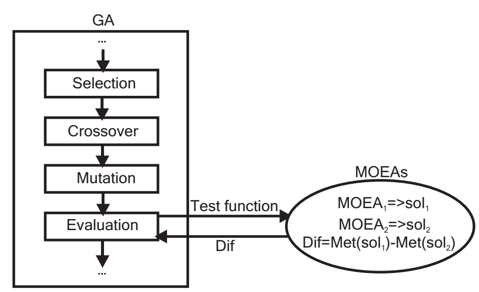

Figure 2. Sketch of the hybrid approach. The quality of a GA chromosome is computed as a difference between the results of applying a performance metric to the solutions computed by two multi-objective evolutionary algorithms (MOEAs).

Because we evolve multi-objective test functions, the optimal solution of such function can be represented as a Pareto front. To compute the performances of the multi-objective algorithms some measures of performance were also introduced. Most of them are applied to the final nondominated set. It is now established that more than one metrics are necessary to evaluate the performances of the multi-objective evolutionary algorithms. Zitzler [18] has recently shown that for an k-objective optimization problem, at least k performance metrics must be used. In our case, we can consider that a convergence metric is sufficient in order to assess and to compare the performance of two multi-objective algorithms.

5.2. Fitness assignment process

We have used the S metric in order to compare the solutions computed by two multi-objective evolutionary algorithms. More exactly, we have computed the metric S for the solution computed by each algorithm A and B - S_A and S_B. The fitness of a GA individual will be the difference between these two area:

- if $S_A \leq S_B$ then the algorithm A has found a better solution than algorithm B
- if $S_A \geq S_B$ then the algorithm B has found a better solution than algorithm A

To avoid the lucky guesses of the optimal points (fronts), each algorithm is run 50 times and the results are averaged. Then, the fitness of a chromosome encoding a test function is computed as the difference between the performances of the two algorithms.

6. Algorithms used for comparison

We describe two multi-objective evolutionary algorithms used for comparison purposes:

- PESA (Pareto Envelope-based Selection Algorithm [9]),
- NSGA II (Nondominated Sorting Genetic Algorithm II [3]).

Apart from standard parameters such as crossover and mutation rates, PESA has two parameters concerning population size, and one parameter concerning the hyper-grid crowding strategy. Therefore, it must be specified the size of the 'internal' population (IP) and the maximal size of the archive, or 'external' population (EP).

Firstly, each chromosome from the 'internal' population is generated and evaluated and EP is initialised to the empty set. Than, the non-dominated members of IP are incorporated into EP. If a termination criterion has been reached, then stop, returning the set of chromosomes in EP as the results. Otherwise, delete the current contents of IP, and repeat the following until $|IP|$ new candidate solutions have been generated: with probability p_c, select two parents from EP, produce a single child. With probability $(1 - p_c)$, select one parent and mutate it to produce a child. Finally, return to incorporate the non-dominated members of IP into EP.

PESA works with a special selection procedure: the mating selection which is based on a crowding measure and it is performed over the archive members. Crowding strategy works by forming an implicit hyper-grid which divides phenotype space into hyper-boxes. Each individual in the archive is associated with a particular hyper-box and has a "squeeze factor" (equal to the number of other individuals from archive which inhabit the same box). Also, environmental selection (archive update) uses the same selection criteria (based on this "crowding measure" for individuals from archive). The difference between environmental and mating selection is made by the selection process that is deterministic, respectively randomized.

In NSGA II the pool of individuals is split into different fronts (each front has assigned a specific rank). All individuals from a front F_i are ordered according to a crowding measure (equal to the sum of distances to the two closest individuals along each objective). The environmental selection is processed based on these ranks. The archive will be formed by the non-dominated individuals from each front (beginning with the best ranking front). Like in PESA, also mating selection is based on the ranking criteria. The new population (obtained after environmental selection) is used for selection crossover and mutation to create a new population. It is important to note that NSGA II uses a binary tournament selection operator but the selection criterion is now based on the crowding comparison operator.

7. Numerical Experiments

Several numerical experiments for evolving multi-objective problems matched to a given algorithm are performed in this section. The algorithms used for comparison have been described in Section 6. Both algorithms work with a binary chromosome representation.

The parameters used by PESA and by NSGA II in order to compute the quality of a GA chromosome are given in Tables 1 and 2.

The parameters of the GA that have been used for evolving multi-objective test functions are given in Table 3.

Table 1. The parameters of the PESA used for numerical experiments

Parameter	Value
Representation	Binary
Internal population size	10
External population size	100
Number of iterations	100
Crossover type	Uniform
Crossover probability	0.7
Mutation type	Strong
Mutation probability	0.05
Grid size (the number of divisions per dimension)	10

We have performed two series of experiments:

- one for comparing the two multi-objective evolutionary algorithms when they work with two objectives ($k = 2$) and
- another experiment for three objectives case ($k = 3$).

The small number of generations (only 10) has been proved to be sufficient for both experiments performed in this paper.

Table 2. The parameters of the NSGA II used for numerical experiments

Parameter	Value
Representation	Binary
Population size	100
Number of generations	100
Crossover type	Uniform
Crossover probability	0.7
Mutation type	Strong
Mutation probability	0.05

Table 3. The parameters of the GA algorithm used for numerical experiments

Parameter	Value
Population size	10
Number of generations	10
Crossover type	Uniform
Crossover probability	0.9
Mutation type	Point mutation
Mutation probability	0.01
Chromosome length	$k \times 16 \times 8$ bits [a]
Runs	30

[a] where k represents the number of objectives

Results are given in Table 4. For each pair of multi-objective algorithms (PESA vs NSGA II and NSGA II vs PESA) is given the average (over 30 runs) of best fitness scored by an individual (encoding a test function) for which one algorithm performs better than the other algorithm.

Table 4. Fitness of the best GA individual in the last generation. Results are averaged over 30 independent runs.

k	PESA vs NSGA	NSGA vs PESA
2	-70.61	-1,169.10
3	-111,372.62	-182,759.43

Table 4 shows that the proposed approach made possible the evolving of multi-objective test functions matched to both given algorithms (all fitness values are negative). The results of these experiments give a first impression of how difficult the problems are. Several interesting observations can be made:

- for both values of objective number (NO), the difference between PESA and NSGA II is smaller than the difference between NSGA II and PESA
- for both comparison (PESA vs NSGA II and, respectively, NSGA II vs PESA) the difference between the two MOEAs increases with the number of objectives

Therefore, the GA algorithm was able to evolve multi-objective functions for which an MO algorithm was better then another MO algorithm.

8. Conclusions and Further Work

In this paper, a framework for evolving multi-objective test problems that are matched to a given algorithm has been proposed. Numerical experiments have shown the efficacy of the proposed approach: multi-objective test problems for which an optimisation evolutionary algorithm performs better than another optimisation evolutionary algorithm have been successfully evolved.

Further research will be focused on the following directions:

- Proving that the evolved test problems are indeed hard for the considered multi-objective evolutionary algorithms.
- Evolving test problems using GP technique (instead of a simple GA). In this case, it is possible to evolve real-valued functions, not only binary ones.
- Finding the set (class) of test problems for which an algorithm is better than the other.

References

[1] D. W. Corne and J. D. Knowles. No Free Lunch and Free Leftovers Theorems for Multiobjective Optimisation Problems. In C. M. Fonseca, P. J. Fleming, E. Zitzler, K. Deb, and L. Thiele, editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, volume 2632 of *LNCS*, pages 327–341. Springer, 2003.

[2] K. Deb. Multi-objective genetic algorithms: Problem difficulties and construction of test problems. *Evolutionary Computation*, 7(3):205–230, 1999.

[3] K. Deb, S. Agrawal, A. Pratab, and T. Meyarivan. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. KanGAL report 200001, Indian Institute of Technology, Kanpur, India, 2000.

[4] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable Test Problems for Evolutionary Multi-Objective Optimization. Technical Report 112, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, 2001.

[5] S. Droste, T. Jansen, and I. Wegener. A natural and simple function which is hard for all evolutionary algorithms. In *GProc. IECON'2000 (IEEE International Conference on Industrial Electronics, Control and Instrumentation)*, pages 2704–2709. IEEE Pres, 2000.

[6] F. Glover. Tabu search– part II. *ORSA Journal on Computing*, 2(1):4–32, 1990.

[7] D. E. Goldberg. *Genetic algorithms in search, optimization and machine learning*. Addison Wesley, 1989.

[8] V. Khare, X. Yao, and K. Deb. Performance Scaling of Multi-objective Evolutionary Algorithms. In C. M. Fonseca, P. J. Fleming, E. Zitzler, K. Deb, and L. Thiele, editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, volume 2632 of *LNCS*, pages 376–390. Springer, 2003.

[9] J. D. Knowles, D. W. Corne, and M. J. Oates. The Pareto-Envelope based Selection Algorithm for Multiobjective Optimization. In *Proceedings of the Sixth International Conference on Parallel Problem Solving from Nature (PPSN VI)*, volume 1917 of *LNCS*, pages 839–848, Berlin, 2000. Springer.

[10] J. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.

[11] M. Oltean. Searching for a practical evidence of the no free lunch theorems. In A. J. Ijspeert, M. Murata, and N. Wakamiya, editors, *BioADIT - Biologically Inspired Approaches to Advanced Information Technology, First International Workshop, BioADIT 2004*, volume 3141 of *LNCS*, pages 472–483. Springer, 2004.

[12] N. J. Radcliffe and P. D. Surry. Fundamental limitations on search algorithms: evolutionary computing in perspective. In J. van Leenwen, editor, *Lecture Notes in Computer Science*, volume 1000 of *LNCS*, pages 275–291. Springer, 1995.

[13] C. Schumacher, M. D. Vose, and L. D. Whitley. The no free lunch and problem description length. In L. Spector and E. D. Goodman, editors, *GECCO 2001: Proc. of the Genetic and Evolutionary Computation Conf.*, pages 565–570. Morgan Kaufmann, 2001.

[14] D. Whitley. Functions as permutations: Implications for no free lunch, walsh analysis and summary statistics. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature – PPSN VI*, LNCS, pages 169–178. Springer, 2000.

[15] D. H. Wolpert and W. G. Macready. No free lunch theorems for search. Technical Report SFI-TR-95-02-010, Santa Fe Institute, 1995.

[16] D. H. Wolpert and W. G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997.

[17] X. Yao, Y. Liu, and G. Lin. Evolutionary Programming made faster. *IEEE-EC*, 3(2):82, 1999.

[18] E. Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, 1999.

[19] E. Zitzler and L. Thiele. Multiobjective Optimization Using Evolutionary Algorithms—A Comparative Study. In A. E. Eiben, editor, *Parallel Problem Solving from Nature V*, LNCS, pages 292–301. Springer, 1998.