Distribution of Populations Generated by Multi-Objective Optimization Evolutionary Algorithms

Przemysław Dziemieszkiewicz[1]

[1] Wroclaw University of Technology,
Institute of Computer Engineering, Control and Robotics, Wroclaw, Poland,
email: pdziemieszkiewicz@lukas.com.pl

Abstract. This paper considers the distribution of populations generated by multi-objective evolutionary algorithms. Some algorithms have a tendency to produce multimodal distributions of population while other algorithms produce a compact cluster of solutions. Particular MOEAs have a tendency to one of these distributions. This paper will show that one feature of an algorithm defines it – the selection method. If selection is based on the evaluation of parents, the algorithm generates a compact population every time. When the selection is based on the fitness of the offspring, the algorithm is able to generate bimodal distribution of population. Five MOEA algorithms were analyzed: VEGA, two types of tournament and two methods proposed by the author (MOEA with competitive selection and MOEA with protective selection). The results of the simulation confirmed that, depending on the selection method selected, the analyzed algorithms generate the expected distribution of population. This paper does not assess which algorithm is better or worse but merely seeks to explain the reason for the algorithms' properties.

1 Introduction

Since 1985 (Shaffer [6]), evolutionary methods have been used for multi-objective optimization problem-solving. The goal of these methods is to find all or a representative group of *Pareto optimal*[1] solutions. However, some algorithms do not have Pareto optimality detection imbedded. These methods, by Fonseca and Fleming called *population-based non-Pareto approaches* [4], need additional procedures. The evolutionary process generates populations of individuals which are monitored by an independent module and Pareto optimal solutions are detected. Pareto optimal solutions could be chosen from the existed populations of individuals only. Therefore the evolutionary process should generate diversified populations, able to produce the full spectrum of potential solutions. Among other things, this ability depends on the distribution of populations. At least two opposite domain-specific problems were observed in this area [4]:

[1] "A point $\vec{x}^* \in \Omega$ is *Pareto optimal* if for every $\vec{x} \in \Omega$ and $I = \{1,2,\ldots,k\}$ either, $\forall_{i \in I}\left(f_i(\vec{x}) = f_i(\vec{x}^*)\right)$ or, there is at least one $i \in I$ such that $f_i(\vec{x}) > f_i(\vec{x}^*)$"
C. A. Coello Coello, D. A. Van Veldhuizen, G. B. Lamont [1].

1. *Speciation*[2] divides population into separated population groups [6]. In this effect, population has multimodal distribution and does not identify compromise solutions.
2. *Premature convergence* reduces the diversity of populations to one compact cluster [2], [5]. As a result, the population has unimodal distribution. Such a compact population usual explores a limited area of search space (local optimum or compromise solutions) and does not explore all Pareto optimal solutions.

The Pareto optimality is not implemented into analyzed algorithms and is not being considered in this paper. Our interest is in the distribution of populations generated by group of multi-objective evolutionary algorithms. Some methods (e.g. as proposed by Shafer VEGA [6]) have a tendency to produce multimodal distribution of population while other algorithms produce unimodal populations. Particular MOEAs have a tendency to one of these adaptations. Three variants of population distribution are being considered (Figure 1.):

1. *Compromise solutions*. A population consisting of solutions with intermediate values of traits suitably adapted to both objectives. The population of such solutions is characterized by unimodal distribution.
2. *Specialized solutions*. Solutions are better adapted to one of the objectives, but with decreased quality for the other. Population is specialized in one objective. The population of such solutions is characterized by unimodal distribution.
3. *Bimodal distribution of solutions*. Solutions are divided into two subpopulations. Each subpopulation is adapted to a different objective. The population of such solutions is characterized by bimodal distribution.

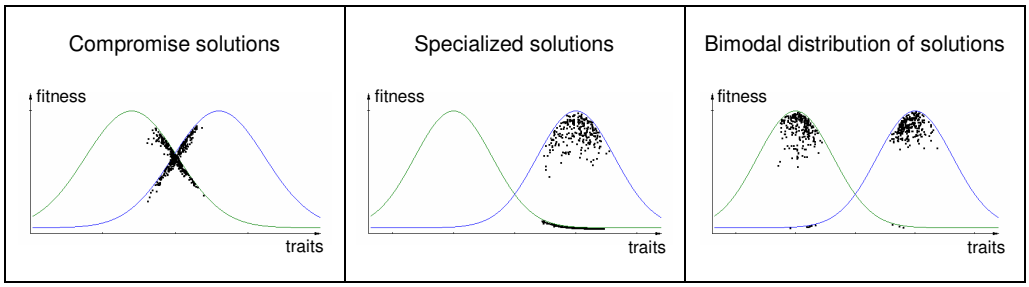

Figure 1. Distribution of populations. All solutions are presented by points, where the ordinate stands for a solution's fitness and the abscissa stands for the position of a solution in the search space.

The previous paper [3] focused on conditions that cause an evolutionary process to lead to varying population distributions in a mosaic environment. A mosaic environment consists of a few niches. The fitness of a solution depends both on a solution's traits (phenotype) and the niche it occupies. The evolution in such an environment could be treated as multi-objective problem-solving. The population is split into subpopulations related to objectives and each solution is evaluated by only one objective function (the same scheme as in the *population-based non-*

[2] Biological term. In MOEA the first time used by Shaffer.

Pareto approaches). It was observed that the distribution of population depends on many factors but one of the most important is the selection method. Two schemes were considered: protective selection and competitive selection. In the table 1., the distribution of population generated by particular selection methods are summarized.

Table 1. Possible distribution of population depending on the method of selection.

kind of selection	compromise solutions	specialized solutions	bimodal distribution of solutions
protective selection	possible	possible	not possible
competitive selection	possible	not possible	possible

The principal difference between selections is that the protective selection evaluates and selects parents, while the competitive selection evaluates and selects offspring that compete for a particular place in the environment. In a homogenous environment, both methods of selection give the same results but in a heterogeneous environment, the offspring may occupy a different niche from its parent and in that case, different criteria of evaluation are applied. In the protective selection, the fitness of an offspring depends on its parent's fitness. If an offspring is assigned to a different objective than the one from which its parents originate, the offspring inherits traits of its parents but not its fitness. The inherited traits may not be suitable for the new local conditions.
When applying competitive selection, we have a different situation. In this case, it is not the parents, but the offspring that are competing for a specific place in the environment which are evaluated. They are evaluated according to the objective attributed to the niche they are competing for. It may happen that some candidates will inherit traits from parents originating from another niche and they will have statistically less chance of surviving selection. In this method, the ability of a solution to survive depends directly on its own (and not its parents) fitness.
These properties are not only restricted to the algorithms mentioned above. There are other methods that are based on the evaluation of parents and later on will be referred to as the *protective type of selection.* Similarly, there are methods which are based on the evaluation of offspring and these will be referred to as the *competitive type of selection*. Comparison of a few algorithms is shown. These examples show that, depending on the type of selection, the multi-objective evolutionary algorithms have defined properties.
This paper does not assess which algorithm is better or worse and the preferred method is not suggested. The sole aim is to explain the reason for the algorithms' properties which could be useful for researchers and engineers.
This paper is organized as follows: the analyzed optimization algorithms are described in the second section, multi-objective problems and simulation results are presented in the third section, and finally, conclusions are given in part four.

2 Multi-objective optimization evolutionary algorithms

The general outline is the same for all analyzed algorithms: The population of M solutions characterized by the vector of N traits. The traits are expressed as real numbers. The succeeding generations are equally numerous and non-overlapping. Each solution of a new generation is an offspring of one of the solutions of the previous generation (asexual reproduction). An offspring inherits slightly modified parental traits. Traits of descendant solutions are determined by adding

to parental traits a random increment with normal distribution N(0, σ). Recombination is not applied. The selection of a solution that will occupy a place in the next generation depends on the selection scheme.
Five multi-objective optimization evolutionary algorithms are compared. Three are competitive types of selection:

1. MOEA-CS (Multi-objective optimization evolutionary algorithm with competitive selection),
2. MOEA-CTS (Multi-objective optimization evolutionary algorithm with competitive type of tournament selection),
3. VEGA (Vector evaluated genetic algorithm)

and two are protective types of selection:

4. MOEA-PS (Multi-objective optimization evolutionary algorithm with protective selection),
5. MOEA-PTS (Multi-objective optimization evolutionary algorithm with protective type of tournament selection).

Algorithms are presented below (the pseudo code is used). Population is split into *P* equally numerous subpopulations. Each subpopulation is bound to one particular objective.

```
01. x[] := new_generation()         // initial generation preparation.
02. REPEAT
03.  FOR p := 1 TO P                // P = number of objectives
                                    // M = population size
04.   FOR i := 1 TO M/P             // M/P = subpopulation size
                                    // group of competitors collection
05.    FOR k := 1 TO K              // K = number of competitors
06.     parent_nr := sampling(x[]) // probability of sampling = 1/M
07.     y[k] := x[parent_nr]        // parent copy
08.     y[k] := modification(y[k]) // competitor modification
09.     f[k] := quality(p, y[k])    // competitor evaluation (objective nr p)
10.    END LOOP
                                    // selection of descendant to the next gen.
11.    x_next_gen[(p-1)*M/P+i] := roulette_selection(y[],f[])
12.   END LOOP
13.  END LOOP

14.  x[] := x_next_gen[]            // exchange of generations
15. UNTIL STOP CONDITION
```

Algorithm 1. Multi-objective optimization evolutionary algorithm with competitive selection (MOEA-CS).

```
01. x[] := new_generation()        // initial generation preparation.
02. REPEAT
                                   // evaluation of parents
03.  FOR p := 1 TO P               // P = number of objectives
04.   FOR i := 1 TO M/P            // M = population size
05.     f[(p-1)*M/P+i] := quality(p, x[(p-1)*M/P+i])  // evaluation of parent
06.   END LOOP
07.  END LOOP
                                   // selection and modification of descendant
```

```
                                  //                  for the next generation
08.  FOR i := 1 TO M              // M = population size
09.   x_next_gen[i] := roulette_selection(x[], f[]) // selection
10.   x_next_gen[i] := modification(x_next_gen[i])  // descendant modification
11.  END LOOP

12.  x[] := x_next_gen[]          // exchange of generations
13. UNTIL STOP CONDITION
```

Algorithm 2. Multi-objective optimization evolutionary algorithm with protective selection (MOEA-PS).

```
01. x[] := new_generation()         // initial generation preparation.
02. REPEAT
03.  FOR p := 1 TO P                // P = number of objectives
                                    // M = population size
04.   FOR i := 1 TO M/P             // M/P = subpopulation size
                                    // group of competitors collection
05.    FOR k := 1 TO K              // K = number of competitors
06.     parent_nr := sampling(x[])  // probability of sampling = 1/M
07.     y[k] := x[parent_nr]        // parent copy
08.     y[k] := modification(y[k])  // competitor modification
09.     f[k] := quality(p, y[k])    // competitor evaluation (objective nr p)
10.    END LOOP
                                    // selec. of descendant to the next gen.
11.    x_next_gen[(p-1)*M/P+i] := tournament_selection(y[], f[])
12.   END LOOP
13.  END LOOP

14.  x[] := x_next_gen[]            // exchange of generations
15. UNTIL STOP CONDITION
```

Algorithm 3. Multi-objective optimization evolutionary algorithm with competitive type of tournament selection (MOEA-CTS). This algorithm is very similar to MOEA-CS. Only the selection function in line 11 (bold type) is different.

```
01. x[] := new_generation()         // initial generation preparation.
02. REPEAT
03.  FOR p := 1 TO P                // P = number of objectives
                                    // M = population size
04.   FOR i := 1 TO M/P             // M/P = subpopulation size
                                    // group of competitors collection
05.    FOR k := 1 TO K              // K = number of competitors
06.     parent_nr := sampling(x[]) // probability of sampling = 1/M
07.     y[k] := x[parent_nr]       // parent copy
08.     f[k] := quality(x[parent_nr].p, y[k]) // competitor evaluation
                                              //        (parent's objective)
09.     y[k] := modification(y[k]) // competitor modification
10.    END LOOP
11.    x_next_gen[(p-1)*M/P+i] := tournament_selection(y[],f[]) // selection
12.   END LOOP
13.  END LOOP

14.  x[] := x_next_gen[]           // exchange of generations
15. UNTIL STOP CONDITION
```

Algorithm 4. Multi-objective optimization evolutionary algorithm with protective type of tournament selection (MOEA-PTS). This algorithm is very similar to MOEA-CTS. Only lines 8 and 9 (bold type) are different.

```
01. x[] := new_generation()       // initial generation preparation.
02. REPEAT
                                  // Calculations for each objective
03. FOR p := 1 TO P               // P = number of objectives
                                  // M = population size
04.  FOR i := 1 TO M/P            // M/P = subpopulation size
05.   y[i] := x[(p-1)*(M/P)+i]    // parent copy
06.   f[i] := quality(p, y[i])    // offspring evaluation (objective nr p)
07.  END LOOP

08.  FOR i := 1 TO M/P            // roulette selection
09.   x_next_gen[(p-1)*(M/P)+i] := roulette_selection(y[], f[])
10.  END LOOP
11. END LOOP

11. FOR i := 1 TO M               // offspring modification
12.  x_next_gen [i] := modification(x_next_gen [i])
13. END LOOP

14.  x[] := shuffle(x_next_gen[]) // shuffling and exchanging of generations
15. UNTIL STOP CONDITION
```

Algorithm 5. Vector evaluated genetic algorithm (VEGA). In the original VEGA the crossover operator was applied as well.

3 Simulation results

The simulations were performed for multidimensional multi-objective problems. These are not real multi-objective problems but a simple example which show the properties of algorithms. The objective functions were constructed with a multidimensional bell-curved Gaussian function:

$$q_p(x_1 \dots x_n) = h_p \cdot e^{-d_p \cdot (x_1 - x_{op})^2} \cdot \prod_{i=2}^{n} e^{-d \cdot x_i^2}, \quad (p = 1,2) \tag{1}$$

The peaks are separated from each other so there are different adaptive optima (x_{01}<x_{02}). The bigger the difference between optima coordinates x_{01} and x_{02} the bigger the diversification of the objectives. The distance between optima Dx is normalized to the standard deviation of mutation and denoted Dx/σ.

During research, simulations with different values of parameters (Dx/σ, M, N, σ, K) were performed. Figure 2. shows representative examples of simulations. All experiments had the same scenario. Simulations began with one compact population located in the middle of the search area (x_{01}= x_{02}= 0). Very quickly (after dozens of generations) the population achieved a quasi-stable distribution which was maintained until the end of the simulation (lasting 10 000 generations).

Histograms (H) and (J) in Fig. 2. present the results of simulations where the population had specialized distribution. The histogram is bimodal because the specialized population was switched between the optimal objectives a few times.

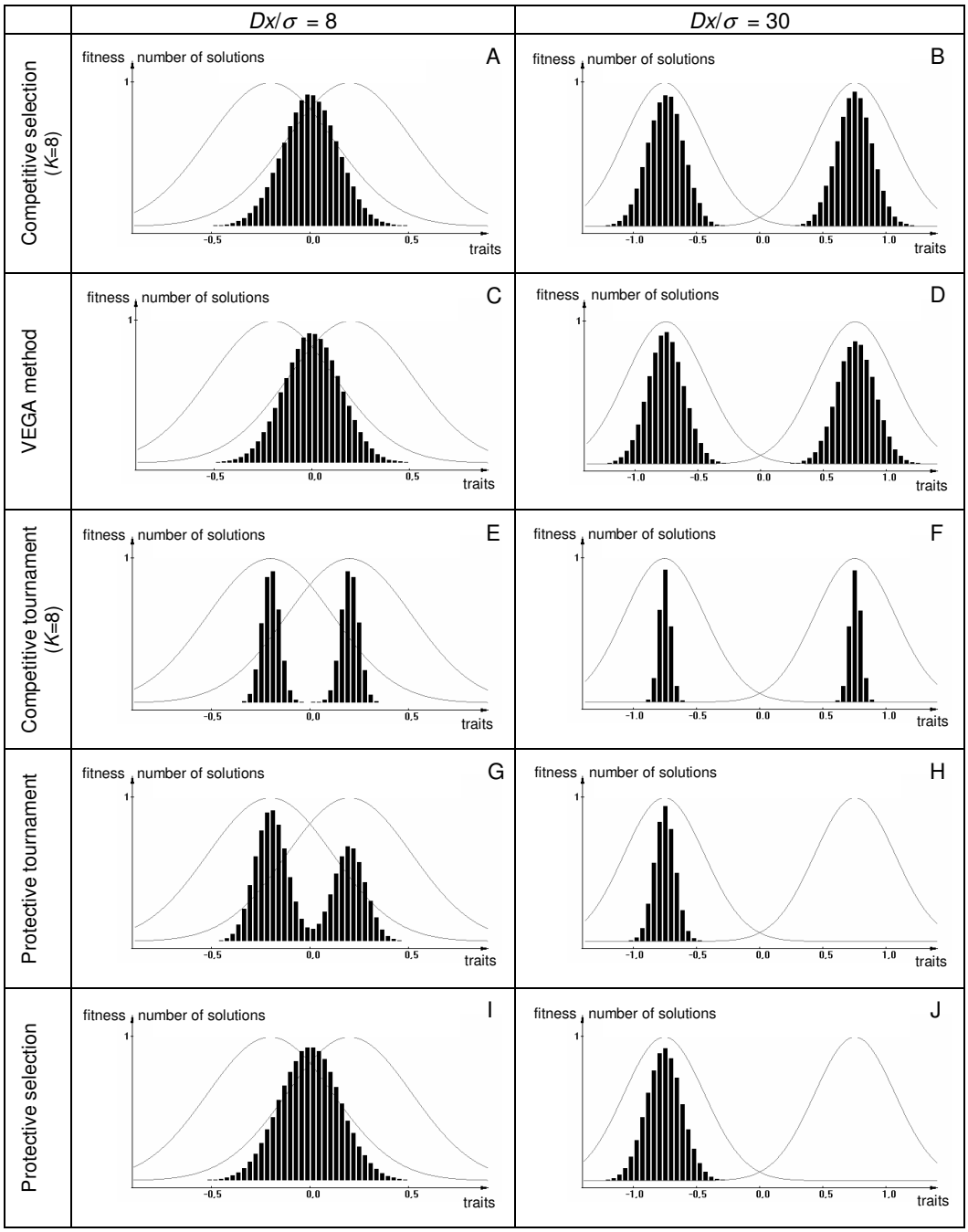

Figure 2. The distribution of population generated by various MOEAs. Histograms of solutions' traits are presented. Parameters of the simulations: M=32; N=4; σ=0,05; h_1=h_2=1; d_1=d_2=d=5.

4 Conclusions

The results of the simulations confirmed that the distribution of population depends on the type of selection algorithm. Those with competitive type of selection (e.g. VEGA, competitive type of tournament, MOEA with competitive selection) are able to generate bimodal distribution of population and these with protective type of selection (e.g. protective type of tournament, MOEA with protective selection) can generate specialized distribution of population (Table 2.). It is impossible to achieve specialized distribution of population using algorithms with a competitive type of selection and it is impossible to generate bimodal distribution of populations using a protective type of selection.

Table 2. Possible distribution of population depending on the method of selection.

MOEA	Type of selection	compromise solutions	specialized solutions	bimodal distribution of solutions
competitive selection	competitive	possible	not possible	possible
competitive type of tournament	competitive	possible	not possible	possible
VEGA	competitive	possible	not possible	possible
protective selection	protective	possible	possible	not possible
protective type of tournament	protective	possible	possible	not possible

Two tournament selections were analyzed. In one criteria problem, both methods generated the same results but for the multi-objective problems, the result depended on the small difference in the selection scheme.

5 References to Literature

[1] C. A. Coello Coello, D. A. Veldhuizen, G. B. Lamont: *Evolutionary Algorithms for Solving Multi-Objective Problems,* Kluwer Academic Publishers, New York, 2002.

[2] A. K. De Jong: An Analysis of the Behavior of a Class of Genetic Adaptive Systems. PhD thesis, University of Michigan.

[3] P. Dziemieszkiewicz, R. Galar: *Simulation of processes of population diversification in heterogeneous environment* – in Polish, X Conference — Simulation of dynamic processes, Zakopane, 1998.

[4] C. Fonseca, P. Fleming: *An Overview of Evolutionary Algorithms in Multiobjective Optimization,* Evolutionary Computation, Vol. 3, No. 3, 165-180, 1995.

[5] D. E. Goldberg, J. Richardson: *Genetic Algorithms with sharing for multimodal function optimization*. red. Grefenstette, 41-49, 1987.

[6] J. D. Schaffer: *Multiple Objective Optimization with Vector Evaluated Genetic Algorithm,* In Genetic Algorithm and their Applications: Proceedings of the First International Conference on Genetic Algorithms, Hillsdale, New Jersey, 93-100, 1985.

[7] http://www.lania.mx/~ccoello/EMOO/EMOObib.html - extensive MOEA bibliography.