

On Gradient-Based and Swarm-Based Algorithms for Set-Oriented Bicriteria Optimization

Wilco Verhoef^(✉), André H. Deutz, and Michael T.M. Emmerich

Multicriteria Optimization, Design and Analytics (MODA) Group, LIACS,
Leiden University, Niels Bohrweg 1, 2375 CA Leiden, The Netherlands
wilco@verhoef.nu, emmerix@gmail.com
<http://moda.liacs.nl>

Abstract. This paper is about the numerical solution of multiobjective optimization problems in continuous spaces. The problem is to define a search direction and a dynamical adaptation scheme for sets of vectors that serve as approximation sets. Two algorithmic concepts are compared: These are stochastic optimization algorithms based on cooperative particle swarms, and a deterministic optimization algorithm based on set-oriented gradients of the hypervolume indicator. Both concepts are instantiated as algorithms, which are deliberately kept simple in order to not obfuscate their discussion. It is shown that these algorithms are capable of approximating Pareto fronts iteratively. The numerical studies of the paper are restricted to relatively simple and low dimensional problems. For these problems a visualization of the convergence dynamics was implemented that shows how the approximation set converges to a diverse cover of the Pareto front and efficient set. The demonstration of the algorithms is implemented in Java Script and can therefore run from a website in any conventional browser. Besides using it to reproduce the findings of the paper, it is also suitable as an educational tool in order to demonstrate the idea of set-based convergence in Pareto optimization using stochastic and deterministic search.

Keywords: Hypervolume indicator · Pareto front · Set-oriented gradient · Particle swarm optimization · Multiobjective optimization · Algorithm animation

1 Introduction

Multi-objective optimization (MOO) is a class of optimization problems where multiple objective functions are optimized simultaneously. MOO problems are common in numerous fields including engineering, science, industry, drug discovery, finance, and logistics. Given this broad range of application areas, there is a big need for fast and reliable MOO algorithms.

In typical MOO problems, the objectives are conflicting. Thus, an optimal solution for one objective is not optimal for the others. Hence, with conflicting objectives, there is no single optimal solution for the problem. Instead, there is typically a whole set of solutions in the decision space that are non-dominated with respect to the Pareto dominance relation. We call this set the *efficient set*. The image of the efficient set obtained by the objective functions is the *Pareto front*. In continuous spaces they can be viewed as a trade-off curve (in 2-D) or a trade-off (hyper)surface (in higher dimensions).

The Pareto front of a function with m objectives is typically a manifold of at most $m - 1$ dimensions and it is not required to be connected. When approximating a Pareto front (and efficient set) by means of a finite approximation set it is a common strategy to search for sets that maximize the size of the dominated (hyper) space which is measured by the hypervolume indicator. In this paper two different strategies for finding hypervolume-maximal sets will be discussed.

For this we will introduce new algorithms for multi-objective optimization, namely a *multi-objective cooperative particle swarm optimization* MOCOPS algorithm and a *multi-objective gradient based optimization* MOGO algorithm (a modified version of a previously discussed set-based gradient strategy). Then we will provide a numerical analysis of the dynamics of these algorithms on bicriteria test problems.

The specific contribution of this research is three-fold. We will analyze the performance of the new algorithmic concepts by using the hypervolume indicator as a quality measure. Secondly this research will focus on an analysis and comparison of the algorithms based on their dynamical visualization (animation). Finally, we will provide a tool that the interested reader can use to further explore the proposed algorithmic concept in an interactive and easy to use web-application.

The structure of the paper as follows: In Sect. 2 we will first introduce the definitions and notation that are to be used in the remainder of the paper. Moreover we will provide a formal definition of the problem and review related work. In Sect. 3 the different multi-objective optimization algorithms are presented. In the Sect. 4 the dynamics of the algorithms on test problems will be compared. We sum up the main findings in Sect. 5 and provide some directions that will be interesting for future work. Instead of extensive statistical plots of repeated runs, we will provide the user with an easy to use javascript program which can be used in a web-browser to reproduce our results and gives the opportunity for self-study of the new algorithms. A description of this visualization tool in Appendices A will conclude the paper.

2 Background

2.1 Definitions and Notation

The space of candidate solutions is called the *decision space*. The space of objective function values with the decision space as domain is called the *objective space*.

For optimization, it is desirable to have an unambiguous way of determining whether an arbitrary vector is considered better than another. For this reason, you can define the relations *weakly-Pareto-dominance* and *strictly-Pareto-dominance* between two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^m$. Weakly-Pareto-dominance is a relation between two vectors where one vector is considered better or equal than another. We will assume that our objective is minimization. Vector \mathbf{x} weakly Pareto-dominates vector \mathbf{y} , if and only if Eq. (1) holds.

$$\forall_{i \in \{1, \dots, m\}} \mathbf{x}_i \leq \mathbf{y}_i \quad (1)$$

A vector \mathbf{x} is considered to *strictly Pareto-dominate* a vector \mathbf{y} if and only if

$$\forall_{i \in \{1, \dots, m\}} \mathbf{x}_i \leq \mathbf{y}_i \wedge \exists_{i \in \{1, \dots, m\}} \mathbf{x}_i < \mathbf{y}_i$$

The strict Pareto-dominance is a strict order where strict dominance of \mathbf{x} over \mathbf{y} is denoted with $\mathbf{x} \prec \mathbf{y}$.

Let $\mathbb{S} \subseteq \mathbb{R}^d$ and $\mathbf{f} : \mathbb{S} \rightarrow \mathbb{R}^m$. In optimization problems \mathbb{S} represents the decision space, and \mathbf{f} the objective functions. The Pareto front of the minimization problem is the non-dominated subset of the image of \mathbb{S} under \mathbf{f} . Formally, we define the Pareto front Y_{PF} in Eq. 2:

$$Y_{PF} := \{\mathbf{f}(\mathbf{x}) \mid \mathbf{x} \in \mathbb{S} \wedge \nexists_{\mathbf{x}' \in \mathbb{S}} \mathbf{f}(\mathbf{x}') \prec \mathbf{f}(\mathbf{x})\} \quad (2)$$

The inverse image of the Pareto set with respect to \mathbf{f} is called the *efficient set*.

The *hypervolume indicator* (HI) is a widely used measure for multi-objective optimization indicating how well the population approximates the Pareto Front [10]. More precisely, it is defined as the Lebesgue measure of the dominated subspace by a population of m -dimensional vectors in the objective space limited by a reference point (to keep it finite), in symbols:

$$HI(P) := \lambda_m \left(\bigcup_{\mathbf{y} \in P} [\mathbf{y}, \mathbf{r}] \right).$$

Here, λ_m denotes the Lebesgue measure in m dimensions, e.g. λ_1 is the length, λ_2 is the area, λ_3 is the volume, and so forth. The reference point is chosen such that it is dominated by all relevant objective vectors. The choice of the reference point should be large enough for it to be dominated by all points that are generated in the optimization.

In this work, we will often consider the contribution of a single point to the dominated hypervolume indicator. The *hypervolume contribution* of a vector in the objective space (objective vector) in a multiset of such vectors (population) is defined as the hypervolume of the total population of objective vectors, minus the hypervolume of the population without that objective vector. Objective vectors which are dominated by some vector in the population have a hypervolume contribution of 0. The concepts of (Pareto) dominance, hypervolume and hypervolume contribution are clarified for two dimensions in Fig. 1.

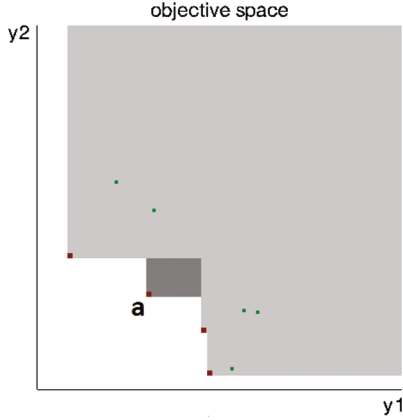


Fig. 1. This is an example of a population in the objective space. Objective vectors in the population are displayed as slightly bigger (as compared to dominated points), black, and filled circles in case they are not dominated by another objective vector in the population. Objective vectors in the population are displayed in dots when they are part of the dominated set. The set of points that is dominated by at least one point in the population is colored light gray. The hypervolume contribution of objective vector *a* is colored dark gray.

In the literature on multiobjective optimization algorithms, different terms with similar meanings are used depending on the field of research. Below is a short summary of interchangeable terms. With Table 1, an attempt has been made to match each field with each of their terms, however when reasoning in general about the algorithms, the terms used might be mixed together. However, we decided to use the terms *population*, *particles*, and *objective functions* to denote the essential entities in the discussed algorithms.

Table 1. Terminology in set-oriented optimization. The terms in bold font will be used throughout this paper.

EA	PSO	Gradient optimization
fitness (function)	fitness (function)	objective function
individual	particle	(search) point, (candidate) solution
population	swarm	approximation set, μd -vector (cf. [6])

Notation	Description
$\mathbb{S} \subseteq \mathbb{R}^d$	Decision space
I	Set of objective vectors, subset of \mathbb{R}^m
m	Dimension of the objective space
d	Dimension of the decision space
$\mathbf{x} \in \mathbb{S}$	Representation of a particle in the decision space
$\mathbf{y} \in \mathbb{R}^m$	Representation of a particle in the objective space
$\mathbf{f} : \mathbb{S} \rightarrow \mathbb{R}^m$	Vector of objective functions
μ	Population size
$\mathbf{a} \in \mathbb{S}$	Particle
$\Phi : \mathbb{S}^\mu \times \mathbb{S} \rightarrow \mathbb{R}$	Fitness contribution of an individual to a population
$HV : I^\mu \rightarrow \mathbb{R}$	Hypervolume indicator of a solution w.r.t. a population of size μ
$\Delta HV : I^\mu \times I \rightarrow \mathbb{R}$	Hypervolume contribution of a single solution in a population
$i \sim u(\{a, b, c\})$	Denotes a uniform random discrete sample $b \in \{a, b, c\}$
$x \sim u([0, 1])$	Denotes a uniform random continuous sample $x \in [0, 1]$
$\mathbf{z} \sim \mathbf{N}(\mathbf{0}, \mathbf{I})$	Denotes a sample from the multivariate i.i.d. normal distribution with mean value 0 and variance 1.

2.2 Problem Definition

For optimization, minimization is assumed for all objectives throughout this work.

$$\mathbf{f}(\mathbf{x}) \rightarrow \min$$

The aspiration of the optimization algorithms in this paper is maximization of the hypervolume indicator over the set of all populations of size μ . That will be the measurement used for benchmarking:

$$HV(P) \rightarrow \max, P \in \mathbb{S}^\mu$$

For this the convergence dynamics of two different types of algorithms will be compared - swarm-based optimization and gradient-based optimization.

2.3 Related Work

The idea to maximize the hypervolume indicator using population based search was initiated first in the context of evolutionary multi-criterion optimization algorithms. For instance, the S-Metric Selection-EMOA (SMS-EMOA) is a popular algorithm in this field [5]. Here the term ‘S-Metric’ is an alternative name for the hypervolume indicator. In particular for small numbers of objective functions ($m = 2, 3$) this algorithm performs very well in comparison with other EAs, although recent research has shown that it does not always converge to the globally Pareto optimal front (which is also the case for most other EMOAs).

Swarm based hypervolume optimization has been suggested before by Mosthaghim et al. [13] in an algorithm oriented at the ‘follow a leader’ paradigm in classical particle swarm algorithm. In our paper we will completely

abandon this and present a cooperative search paradigm where each particle can contribute in equal proportions to the success of a population.

Especially in the ‘fine tuning’ phase of the set-based optimization on differentiable problems it is not very promising to use stochastic search algorithms with isotropic mutation operators. This is because the local tangent cone in which dominating solutions of a point can be found in the efficient space gets increasingly acute and the opening angles converge to zero. This is why sampling steps must be exactly in descent directions in order to get closer to the Pareto front. This is where gradient-based search presents itself as an interesting alternative to stochastic search.

Research about gradient optimization algorithms in MOO using the hypervolume indicator has been performed before [6]. In the research, computation complexity of the algorithms is reported for a varying numbers of dimensions. It was shown by Emmerich et al. [6] that essentially the hypervolume contribution gradients are the sub-gradient components of the entire population vector. While [6] focuses on the computation of the Hypervolume gradient and its general properties, its search dynamics were investigated on a couple of low dimensional problems in [6]. Here a simple set-based gradient and set-based Newton method was compared on bi-criterion problems. In our paper we will study an improved version of this algorithm that introduces a non-zero gradient for dominated points.

3 Optimization Algorithms

In this study we consider iterative (sequential) algorithms for Pareto optimization. They generate a series of populations P_0, P_1, \dots , that (probabilistically) converge to the Pareto front. Such processes will be visualized and studied using a web-based interface.

On the website <http://moda.liacs.nl/pareto-demo.html> the two algorithms under investigation are implemented in `java script`, an interpreter language which features client side execution. The featured algorithms are a cooperative swarm based algorithm and a hypervolume gradient method.

As opposed to single objective optimization, in multiobjective optimization instead of a starting point a starting population needs to be provided for iterative optimization algorithms. Alternatively, we may initialize the population uniformly randomly within the search space or on points of a regular grid.

3.1 Multi-objective Particle Swarm Optimization Algorithm

In the interpretation we use in this paper a particle swarm optimization (PSO) algorithm is a randomized search heuristics where a swarm of particles moves gradually towards an optimal solution driven by randomized modification operators and interaction between the particles.

In conventional PSO algorithms, the swarm is driven by a leader, who is the currently best individual in a population, and by local memories of particles on

their so-far best positions. In single-objective optimization such processes will typically converge to local, or sometimes even to global optima. In multiobjective optimization such an approach could be easily used to find a single point on the Pareto front, but is not well suited to distribute points across the Pareto front, because the particles all strive to resemble the leader which is counter-productive when searching for a diverse set of solutions. To a certain extent this can be compensated by assigning local leaders, but this makes the algorithm quite complicated and adds parameters to the algorithm (i.e., number of leaders).

The use of traditional PSO for multi-objective optimization problems has been addressed already in the literature, both in the context of general multiobjective optimization [4], and for finding Pareto fronts that maximize the hypervolume indicator [13]. Both approaches let to algorithms that can produce good approximations to Pareto fronts.

In this paper, however, we consider another approach that we will term *cooperative particle swarm*. This algorithm will have the following properties that distinguishes it from previous swarm-based approaches:

- Leader-free: The particles in the population cooperate in covering the Pareto front, instead of competing with each other. There is no leader in the swarm; each particle strives to contribute to the global performance of the swarm.
- Indicator-based: The algorithms seeks to maximize an unary performance indicator. Here the hypervolume indicator is used, but also other unary indicators could be considered (e.g., *reference point free hypervolume* [8]).

The new approach is deliberately kept very simple. This is for two reasons: Firstly we want to demonstrate that only a few essential components are needed to steer a swarm towards a Pareto front. Secondly, simplicity will make the algorithms easier accessible to a rigorous theoretical analysis. It will also be easier to compare it on a conceptual level to the later discussed set-gradient based algorithm.

We will term the approach Multiobjective Optimization by Cooperative Particle Swarms (MOCOPS).

The pseudo-code for the proposed MOCOPS algorithm is given in Algorithm 1. It starts with randomly initializing a set of particles. Then, in each iteration of the algorithm, a particle is randomly selected and a small random variation of this particle is generated by adding a vector of normally distributed random numbers.

If the fitness contribution of the mutated particle relative to the population is better than for the original position then the particle will move to the new position: Firstly, it will be tested which one of the two positions leads to a better hypervolume indicator of the population. Secondly, if both positions are equally good (which will typically occur for dominated solutions), the point that has a better value in the aggregated linear objective function with equal weights is considered. Note that if one solution is dominated by the other solution it will also be considered better in the latter comparison (because of positive equal weighting). Therefore, eventually all solutions will strive towards the non-dominated front and then their hypervolume contribution will be considered.

The cycle continues with picking a random particle again. The MOCOPS algorithm is displayed in pseudocode in Algorithm 1. Care must be taken to ensure $\mathbf{x}_{new} \in \mathbb{S}$ (e.g. by rejecting infeasible vectors).

Algorithm 1. Multiobjective Optimization by Cooperative Swarms (MOCOPS)

```

Input initial population  $P_0$ 
while termination criterion is not reached do
   $t \leftarrow t + 1$ 
   $s \sim u(\{1, 2, \dots, n - 1, n\})$ 
   $\mathbf{x}_{old} = \mathbf{x}^{(s)}$ 
   $P \leftarrow P_t \setminus \{\mathbf{x}^{(s)}\}$ 
  {Try to improve position of particle  $\mathbf{x}^{(s)}$ }
   $\mathbf{z} \sim \mathbf{N}(\mathbf{0}, \mathbf{I})$ 
   $\mathbf{x}_{new} = \mathbf{x}_{old} + \sigma \cdot \mathbf{z}$ 
  if  $HV(P \cup \{\mathbf{x}_{new}\}) > HV(P \cup \{\mathbf{x}_{old}\})$  then
     $P_t \leftarrow P \cup \{\mathbf{x}_{new}\}$ 
  else if  $HV(P \cup \{\mathbf{x}_{new}\}) < HV(P \cup \{\mathbf{x}_{old}\})$  then
     $P_t \leftarrow P \cup \{\mathbf{x}_{old}\}$ 
  else if  $f_1(\mathbf{x}_{new}) + f_2(\mathbf{x}_{new}) < f_1(\mathbf{x}_{old}) + f_2(\mathbf{x}_{old})$  then
     $P_t \leftarrow P \cup \{\mathbf{x}_{new}\}$ 
  else
     $P_t \leftarrow P \cup \{\mathbf{x}_{old}\}$ 
  end if
end while
Return  $P_t$ 

```

One iteration of the bicriteria MOCOPS algorithm can be performed with a time complexity in $\mathcal{O}(\mu \log \mu)$ and its complexity is related to the problem of computing the hypervolume contribution of a point which is discussed in [9]. However, by implementing the algorithm as an online algorithm, that is using incremental update steps, we can compute a single iteration with time complexity in $\mathcal{O}(\log \mu)$ (amortized over the number of iterations) [12]. Fast - linear time - hypervolume update schemes are also known for three objective functions [11]. The computational complexity is expected to grow exponentially in the number of objective functions [3], that is why the scheme does probably not lend itself very well for many-objective optimization.

3.2 Adaptive Mutation

A weakness of the algorithm design of Algorithm 1 is that the particles are always perturbed with the same distribution and average step-size. When a particle is far away from the optimum a relative big step size will be beneficial. When the solution is already very close to an optimal position then fine-tuning is required, and thus smaller steps.

To account for this we introduce a simple scheme for the adaptation of the standard deviation. In earlier research it has been found that adapting the mutation rate in a stochastic descend method a rate of $\frac{1}{5}$ is a good heuristic choice [1, 2]. This has been called the $1/5^{th}$ *success rule*. In our approach we multiply σ or divide it by a scalar in order to keep the success rate of trial moves approximately $\frac{1}{5}$. Heuristically this scalar has been determined as $\sqrt[4]{1.04}$.

In earlier research it has been argued that self-adaptive step-size control does not work in the context of multiobjective optimization, because the boundary between the region where a point improves (Pareto dominates the original point) and the region where a point does not improve becomes cusp-like. However, in the context of hypervolume maximization the boundary between these two spaces is, in most relevant cases, differentiable. This means that in the limit of an infinitely small step-size always a success rate of $1/2$ can be achieved, unless the point is exactly on its optimal position on the Pareto front. This is because the improvement region and non-improvement regions are locally separated by a $m - 1$ dimensional hyperplane, and the probability to generate a trial point on either one side of the plane is the same.

3.3 Multi-objective Gradient Based Optimization Algorithm

The MOGO algorithm is a deterministic algorithm where each search point is simultaneously directed by a search point dependent subgradient of the total hypervolume gradient. For example, the search point $\mathbf{x}^{(i)}$ could be directly directed by the gradient of \mathbf{f} . This will however lead to the convergence to a local optimum. However, it is expected that using the gradient this way will lead to little diversification. Therefore, instead we will use a search direction that is derived from the gradient of the entire population, which was derived in [7] and discussed in more detail in [6]. This strategy leads to convergence as well as to diversification (see for instance [14]).

In [6] it was shown that following the set-gradient of the hypervolume indicator is equivalent to simultaneously moving the particles of a population in a direction of steepest descent of their hypervolume contributions to the population. This direction we will denote with $\nabla\Delta HV(\mathbf{x}, P)$ for some particle $\mathbf{x} \in P$.

The MOGO algorithm we will propose next will basically follow the lines of the gradient flow algorithm in [14], but some important modification will be made in order to eliminate problems with losing dominated solutions and ‘creepyness’ – a term used in [14] to describe problems caused by large differences in the length of local gradient components.

We use the same definition of $\nabla\Delta HV(\mathbf{x}, P)$ as in [6] and, for the sake of brevity, restrict our discussion on the bicriteria case.

Let $\mathbf{f} : \mathbb{R}^d \rightarrow \mathbb{R}^2$ denote the vector valued objective function. The Jacobian Matrix of \mathbf{f} at \mathbf{x} is shown below¹.

$$\mathbf{J}_{\mathbf{f}}(\mathbf{x}) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(\mathbf{x}) & \cdots & \frac{\partial f_1}{\partial x_d}(\mathbf{x}) \\ \frac{\partial f_2}{\partial x_1}(\mathbf{x}) & \cdots & \frac{\partial f_2}{\partial x_d}(\mathbf{x}) \end{pmatrix} \quad (3)$$

The values in a Jacobian indicate how the objective function values respond on a change of the input. The hypervolume contribution derivative vector for the mapping $\mathbb{R}^2 \rightarrow \mathbb{R}_+^0$ is defined below:

$$\nabla HV(\mathbf{y}) := \begin{pmatrix} \frac{\partial \Delta HV}{\partial y_1} \\ \frac{\partial \Delta HV}{\partial y_2} \end{pmatrix} \quad (4)$$

Its computation can be done by computing the lengths of the line segment in the attainment surface (or ‘staircase’) that are adjacent to $\mathbf{f}(\mathbf{x})$. For a derivation, see [6].

Multiplication of the matrix and vector yields a gradient of the ΔHV .

$$\nabla \Delta HV(\mathbf{x}, P) = (\mathbf{J}_{\mathbf{f}}(\mathbf{x})^\top \cdot \nabla HV(\mathbf{f}(\mathbf{x})))^\top. \quad (5)$$

In [14] an algorithm was studied that follows the flow of this gradient. It was found to suffer from a strong discrepancy in the length of the subgradients for different points which led to problems in convergence. In this study we counteract this problem by using instead of $\nabla HV(\mathbf{y})$ the normalized subgradient $\nabla HV(\mathbf{y})/||\nabla HV(\mathbf{y})||$:

$$\nabla_N \Delta HV(\mathbf{x}) := \left(\mathbf{J}_{\mathbf{f}}(\mathbf{x})^\top \cdot \frac{\nabla HV(\mathbf{f}(\mathbf{x}))^\top}{||\nabla HV(\mathbf{f}(\mathbf{x}))||} \right). \quad (6)$$

This way the gradient direction does not change, but the difference in length of sub-gradients is compensated for. Yet, the length of the total gradients decreases in the proximity of hypervolume maximal solutions, which is desirable and makes an additional step-size adaptation mechanism no longer needed.

Moreover, in order to not ‘lose’ points that are Pareto dominated within the population – they have zero gradients – we propose to move them in the following direction, which is guaranteed to point into the dominance cone in case of locally non-dominated solutions

$$\nabla \left(-\frac{1}{2}f_1 - \frac{1}{2}f_2 \right) (\mathbf{x}) = -\frac{1}{2} (\nabla f_1(\mathbf{x}) + \nabla f_2(\mathbf{x})). \quad (7)$$

Instead of $\frac{1}{2}$, it is also possible to use a small positive step-size σ . This will be done here and which also stresses the analogy to the step-size used in the swarm based search.

¹ In this paper we restrict ourselves to bicriteria optimization but the introduced principles are applicable also in higher dimensions.

The MOGO algorithm follows directly from these preliminaries. It starts with initializing a randomly distributed population. For every search point the descent vector is computed and added to the current solution. The algorithm terminates if stagnation sets in because all descent directions are zero vectors. The MOGO algorithm is displayed in Algorithm 2.

Algorithm 2. Multiobjective Gradient-based Optimization (MOGO)

Initialize randomly distributed population

$P_0 = (\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(\mu)})$

repeat

$c \leftarrow 0$

for $i \in \{1, \dots, \mu\}$ **do**

if $\|\nabla \Delta HV(\mathbf{x}^{(i)})\| \neq 0$ **then**

$\mathbf{q} \leftarrow \mathbf{x}^{(i)} + \sigma \cdot \nabla_N \Delta HV(\mathbf{x}^{(i)})$

else if $\nabla(-f_1 - f_2)(\mathbf{x}) \neq 0$ **then**

$\mathbf{q} \leftarrow \mathbf{x}^{(i)} + \sigma \nabla(-f_1 - f_2)(\mathbf{x})$

else

$\mathbf{q} \leftarrow \mathbf{x}^{(i)}$

end if

$\mathbf{x}^{(i)} \leftarrow \mathbf{q}$

$c \leftarrow c + 1$

end for

until $c = \mu$

The computational time complexity of the MOGO algorithm is determined by the time complexity for computing the full hypervolume gradient, the components of which are used as descent directions. It was shown in [6] to be in $\Theta(d\mu + \mu \log \mu)$ (provided the number of objective functions is lower than 4).

4 Evaluation

4.1 Test Problems

Problem 1. The objective functions for problem 1 are depicted below. The reference point used for the hypervolume indicator will be the maximal point $(1.25, 1.25)$ (Fig. 2).

$$f_1(\mathbf{x}) = (x_1)^2 + (x_2 - 0.5)^2 \quad (8)$$

$$f_2(\mathbf{x}) = (x_1 - 1)^2 + (x_2 - 0.5)^2 \quad (9)$$

Problem 2. The objective functions for problem 2 are depicted below. The reference point used for the hypervolume indicator will be the maximal point $(1, 1)$ (Fig. 3).

$$f_1(\mathbf{x}) = 1 - ((x_1)^2 + 1)(x_2)^2 \quad (10)$$

$$f_2(\mathbf{x}) = 1 - ((x_2)^2 + 1)(x_1)^2 \quad (11)$$

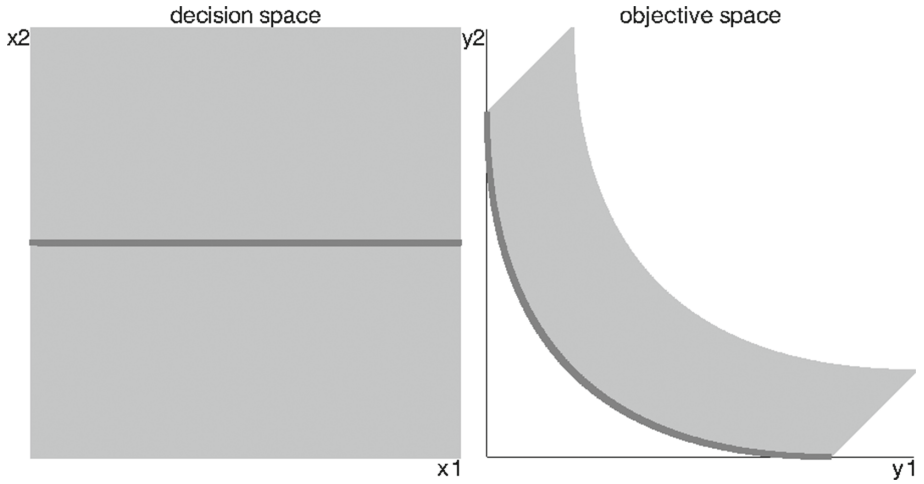


Fig. 2. The decision space and objective space of test problem 1 are shown using a uniformly distributed population of 250000 particles. The area covered with dark gray particles denote the efficient and Pareto set among the population. The area covered with light gray particles resembles the subset which is dominated by the Pareto set of the population. The efficient set is horizontal.

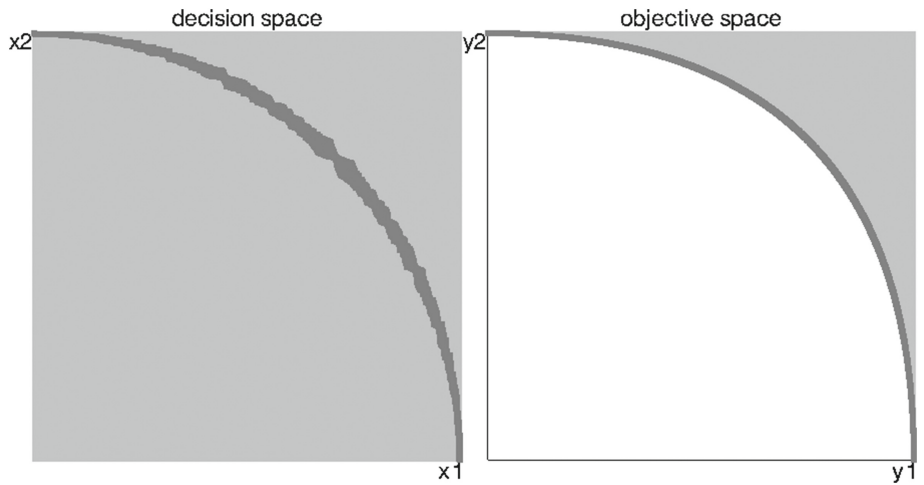


Fig. 3. The decision space and objective space of test problem 2 are shown using a uniformly distributed population of 250000 particles. The area covered with dark gray particles denote the efficient and Pareto set among the population. The area covered with light gray particles resembles the subset which is dominated by the Pareto set of the population. The efficient set is curved.

4.2 Experiments Setup

In this paper the number of reported experiments is kept relatively small. Instead of focussing on statistics we invite the reader to experiment with the implementation which is made available on the website <http://moda.liacs.nl/pareto-demo>.

With $\mu = 100$ we are going to compare the different algorithms based on the hypervolume indicator with varying amount of iterations. The amount of iterations that will be used is 10, 100 and 1000. Each of these results will be sampled from 100 tests. The 10, 100 and 1000 iterations tests will be performed on both test problems. The MOCOPS will be benchmarked with and without adaptive mutation. The MOCOPS algorithms will have their mutation rate initialized with 0.2. The step size of the MOGO is initialized with 0.0008.

Also we will have a look at the dynamics of the algorithms in a visual way. We will compare the converged populations of the different algorithms.

4.3 Description of Results

The results of the benchmark on problem 1 and 2 can respectively be found in Tables 2 and 3. MOCOPS is the MOCOPS algorithm without adaptive mutations and MOCOPS A is the MOCOPS algorithm with adaptive mutation.

In Figs. 4 and 5 the converged populations of respectively the MOCOPS and MOGO algorithm are shown of test problem 2. It seems the algorithms converge with a slightly different alignment. Both alignments are diverse and approximate the real Pareto set well.

4.4 Discussion of Results

Looking at the benchmarks, it seems the performance of the algorithms is more or less the same. The MOGO seems to perform slightly better than the MOCOPS

Table 2. Test problem 1

Algorithm	10	100	1000
MOCOPS	1.3602	1.3713	1.3879
MOCOPS adaptive	1.3609	1.3696	1.385
MOGO	1.3632	1.3812	1.3909

Table 3. Test problem 2

Algorithm	10	100	1000
MOCOPS	1.3604	1.3709	1.3879
MOCOPS adaptive	1.3606	1.37	1.3851
MOGO	1.3639	1.3807	1.3909

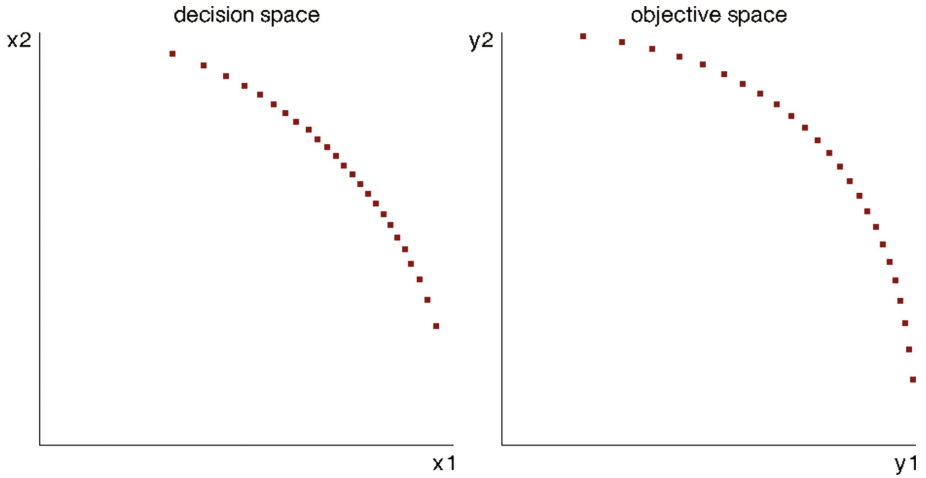


Fig. 4. A screenshot of the converged population with the MOCOPS algorithm on test problem 2.

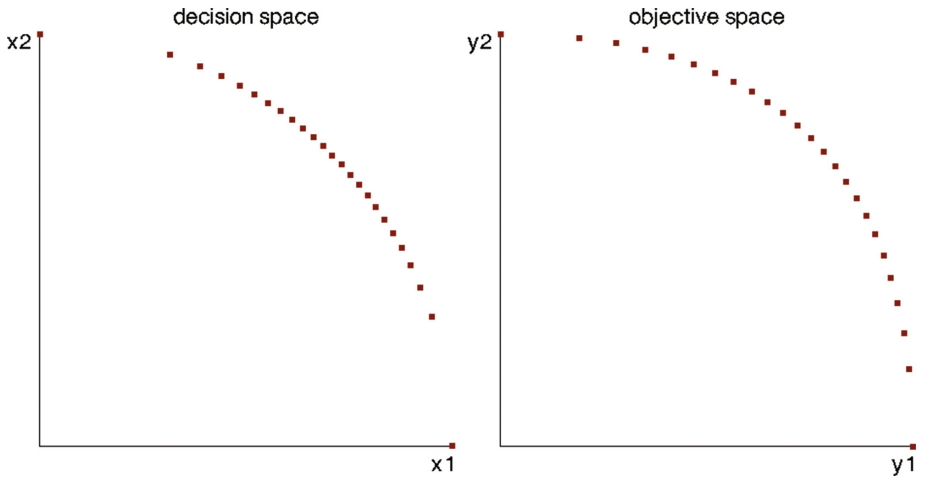


Fig. 5. A screenshot of the converged population with the MOGO algorithm on test problem 2.

variants. The visualization seems to offer more convenient information. Experimenting with the algorithms show that nearly the whole population quickly finds a spot on the Pareto front with the MOGO algorithm. In the MOCOPS algorithm some particles tend to lag behind. Since the MOGO algorithm updates the whole population simultaneously, it is much faster with larger populations.

5 Conclusion

It turns out both the algorithms perform well on the test problems in the sense that they are capable to deliver precise approximations of the Pareto front.

However, the MOGO algorithm outperforms the MOCOPS algorithm. This especially holds with a big population. The MOGO tends to get a nearly perfect diversification. Notably, using the adaptations (gradient length normalization, using the direction as in (7) for dominated points) that were introduced in this paper the MOGO algorithms does not exhibit the problems that were observed for earlier hypervolume-gradient descent schemes. At the same time, it needs to be remarked here that the MOCOPS approach is more robust and does not require differentiability. It is also more promising, when multimodal problems should be considered, though further adaptation would certainly be required for a competitive performance in the multimodal case as compared to other state-of-the-art strategies developed for this problem domain.

For future research it will be interesting to see how the MOCOPS will perform when the adaptive mutation is performed individually instead of globally. The analysis in this paper was on very simple optimization problems and at most can serve as a proof of concept study. Experiments with more challenging test problems and comparisons to state-of-the-art methods will be required before the strategies proposed in this paper can be recommended for practical usage. Based on the first results, we think both approaches to be interesting approaches for further assessment and development.

A Appendices

A.1 Manual of the Application

The application and code is available online [15]. Using the application is straightforward and does not need any installation or configuration. The application can be started with any modern browser. The application is shown in Fig. 6.

1. This is the sidebar with the interaction parameters.
 - (a) Pressing the *Printable* button sets the background color to white. Pressing the *Regular* button will set the color scheme regular again.
 - (b) In the problem section you can choose a test problem.
 - (c) In the initialization section you can select the population size. You can initialize the population with the set population size by pressing one of the buttons. Pressing the *Initialize randomly* button will position the particles at random in the decision space. Pressing the *Initialize uniformly* button will try to position the particles uniformly in the decision space.
 - (d) In the algorithm section you can choose the optimization algorithm by pressing *Particle swarm optimization* or *Gradient based optimization*. Deselecting the *Enable dominated set* will enable the use of the whole population. Pressing *Adaptive mutation* makes the MOCOPS algorithm

use the $1/5^{th}$ success rule. The mutation rate for the MOCOPS algorithm can be adjusted by using the slider. The step size of the MOGO can be adjusted similarly with the other slider.

- (e) In the optimization section you can select how many milliseconds delay every iteration should have using the slider. A bigger delay can make the dynamics of the algorithm clearer. The button *Start* will start the selected algorithm. Pressing the button *Stop* will stop the algorithm again. The *Benchmark* button will run some benchmarks and outputs the statistics in

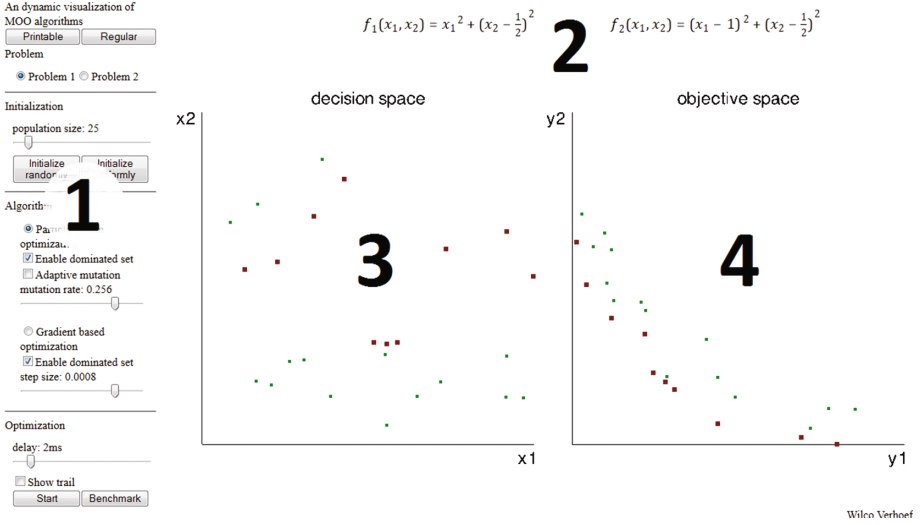


Fig. 6. A screenshot of the interactive multi-objective optimization application.

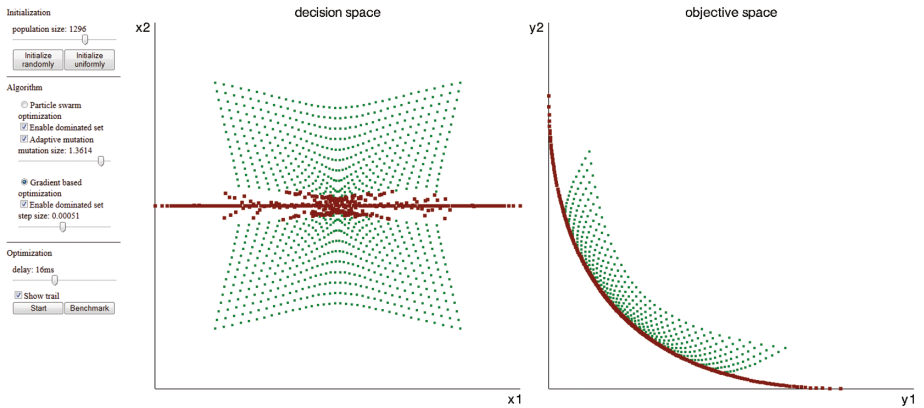


Fig. 7. A screenshot during convergence of the population with the MOGO algorithm on test problem 1.

the browser console. In most browsers the console is accessible by pressing F12.

- (f) The application automatically adapts to the window size. Full screen mode is available with F11.
2. The objective functions of the chosen test problem are shown here.
3. This part of the screen shows the decision space. Points of the efficient set of the population are displayed in red squared dots. The particles which are dominated by the particles in the efficient set are displayed as green dots.
4. This part of the screen shows the objective space. Points of the Pareto set of the population are displayed in red squared dots. The particles which are dominated by the Pareto set are shown as green dots.

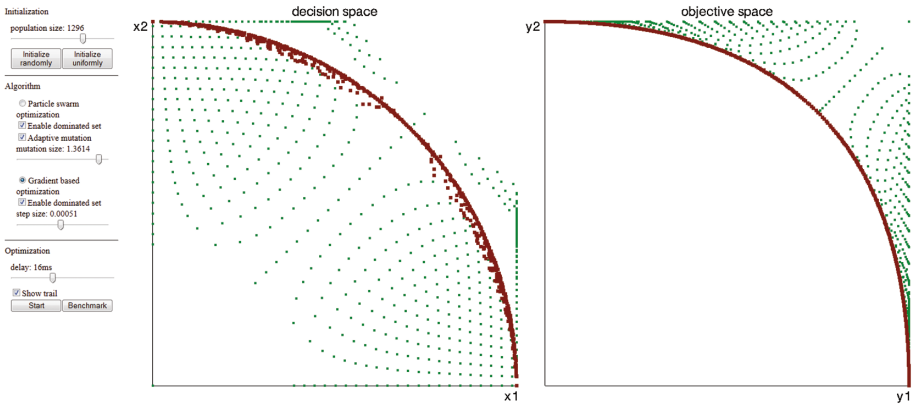


Fig. 8. A screenshot during convergence of the population with the MOGO algorithm on test problem 2.

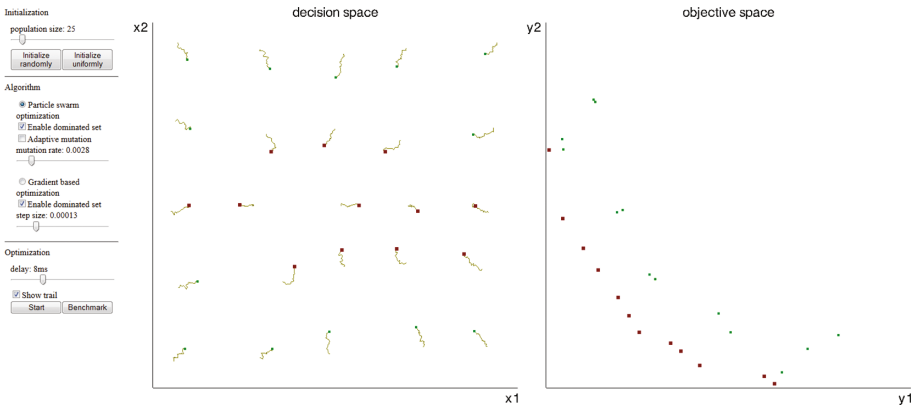


Fig. 9. A screenshot with path tracing during the convergence of a population with the MOCOPS algorithm on test problem 1.

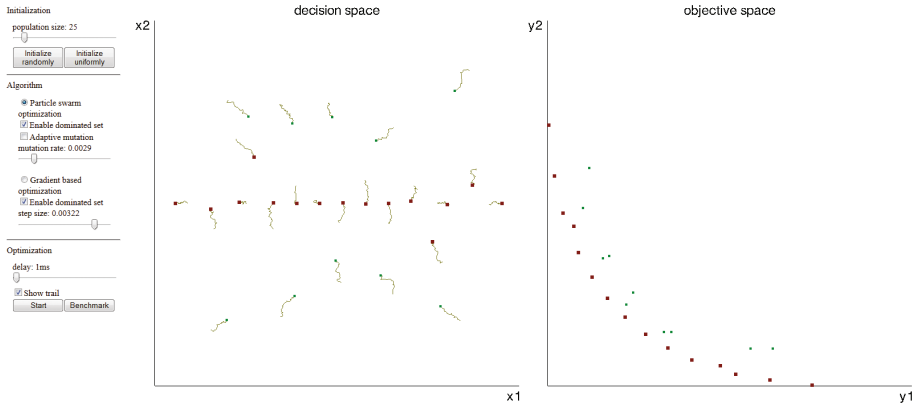


Fig. 10. A screenshot with path tracing during the convergence of a population with the MOCOPS algorithm on test problem 1 at a late stage.

Finally, we exhibit some example screenshots:

- Figure 7 shows a large population that is in the process of converging towards the Pareto front starting from a uniformly distributed sample. The MOGO algorithm is applied on Problem 1. Green, small points are dominated and red squared dots are non-dominated with respect to the other points in the population.
- Figure 8 shows a large population that is in the process of converging towards the Pareto front starting from a uniformly distributed sample. The MOGO algorithm is applied on Problem 2.
- Figure 9 shows a small population of particles moved by the MOCOPS algorithm on Problem 1. Also, the traces of the recent moves of the particles are visualized. Note, that points on the efficient set move sideways in order to find their optimal position with respect to diversity (hypervolume contribution).
- Figure 10 shows the same population as shown in Fig. 9 at a later stage of the convergence process.

References

1. Auger, A.: Benchmarking the (1+1) evolution strategy with one-fifth success rule on the bbob-2009 function testbed. In: Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers, GECCO 2009, pp. 2447–2452. ACM, New York (2009)
2. Beyer, H.-G., Schwefel, H.-P.: Evolution strategies—a comprehensive introduction. *Nat. Comput.* **1**(1), 3–52 (2002)
3. Bringmann, K., Friedrich, T.: Approximating the least hypervolume contributor: Np-hard in general, but fast in practice. In: Evolutionary Multi-Criterion Optimization, pp. 6–20. Springer (2009)

4. Coello Coello, C.A., Lechuga, M.S.: Mopso: a proposal for multiple objective particle swarm optimization. In: *Proceedings of the 2002 Congress on Evolutionary Computation, CEC 2002*, vol. 2, pp. 1051–1056. IEEE (2002)
5. Emmerich, M., Beume, N., Naujoks, B.: An emo algorithm using the hypervolume measure as selection criterion. In: *Evolutionary Multi-Criterion Optimization*, pp. 62–76. Springer (2005)
6. Emmerich, M., Deutz, A.: Time complexity and zeros of the hypervolume indicator gradient field. In: Schuetze, O., Coello Coello, C.A., Tantar, A.-A., Tantar, E., Bouvry, P., Del Moral, P., Legrand, P. (eds.) *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation III. Studies in Computational Intelligence*, vol. 500, pp. 169–193. Springer International Publishing (2014)
7. Emmerich, M., Deutz, A., Beume, N.: Gradient-based/evolutionary relay hybrid for computing Pareto front approximations maximizing the S-metric. Springer (2007)
8. Emmerich, M.T.M., Deutz, A.H., Yevseyeva, I.: On reference point free weighted hypervolume indicators based on desirability functions and their probabilistic interpretation. *Procedia Technol.* **16**, 532–541 (2014)
9. Emmerich, M.T.M., Fonseca, C.M.: Computing hypervolume contributions in low dimensions: asymptotically optimal algorithm and complexity results. In: *Evolutionary Multi-Criterion Optimization*, pp. 121–135. Springer (2011)
10. Fleischer, M.: The measure of pareto optima applications to multi-objective metaheuristics. In: *Evolutionary Multi-Criterion Optimization*, pp. 519–533. Springer (2003)
11. Guerreiro, A.P., Fonseca, C.M., Emmerich, M.T.M.: A fast dimension-sweep algorithm for the hypervolume indicator in four dimensions. In: *CCCG*, pp. 77–82 (2012)
12. Hupkens, I., Emmerich, M.: Logarithmic-time updates in sms-emoa and hypervolume-based archiving. In: *EVOLVE-A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation IV*, pp. 155–169. Springer (2013)
13. Mostaghim, S., Branke, J., Schmeck, H.: Multi-objective particle swarm optimization on computer grids. In: *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, GECCO 2007*, pp. 869–875. ACM, New York (2007)
14. Hernández, V.A.S., Schütze, O., Emmerich, M.: Hypervolume maximization via set based Newton’s method. In: Tantar, A.-A., Tantar, E., Sun, J.-Q., Zhang, W., Ding, Q., Schtze, O., Emmerich, M., Legrand, P., Del Moral, P., Coello Coello, C.A. (eds.) *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation V. Advances in Intelligent Systems and Computing*, vol. 288, pp. 15–28. Springer International Publishing (2014)
15. Verhoef, W.: Interactive demo on multi-objective optimization, liacs, leiden university, nl, wilco.verhoef.nu/projects/moo, bsc project (2015)

EVOLVE - A Bridge between Probability, Set Oriented
Numerics, and Evolutionary Computation VI

Tantar, A.-A.; Tantar, E.; Emmerich, M.; Legrand, P.;
Alboaie, L.; Luchian, H. (Eds.)

2018, XIV, 226 p. 84 illus., Softcover

ISBN: 978-3-319-69708-6