Applied Soft Computing 33 (2015) 223–230

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Seeding the initial population of multi-objective evolutionary algorithms: A computational study

Tobias Friedrich [a], Markus Wagner [b,*]

[a] Hasso Plattner Institute, Potsdam, Germany
[b] University of Adelaide, Australia

ARTICLE INFO

Article history:
Received 25 November 2014
Received in revised form 21 April 2015
Accepted 22 April 2015
Available online 30 April 2015

Keywords:
Multi-objective optimization
Approximation
Comparative study
Limited evaluations

ABSTRACT

Most experimental studies initialize the population of evolutionary algorithms with random genotypes. In practice, however, optimizers are typically seeded with good candidate solutions either previously known or created according to some problem-specific method. This *seeding* has been studied extensively for single-objective problems. For multi-objective problems, however, very little literature is available on the approaches to seeding and their individual benefits and disadvantages. In this article, we are trying to narrow this gap via a comprehensive computational study on common real-valued test functions. We investigate the effect of two seeding techniques for five algorithms on 48 optimization problems with 2, 3, 4, 6, and 8 objectives. We observe that some functions (e.g., DTLZ4 and the LZ family) benefit significantly from seeding, while others (e.g., WFG) profit less. The advantage of seeding also depends on the examined algorithm.

1. Introduction

In many real-world applications trade-offs between conflicting objectives play a crucial role. As an example, consider engineering a bridge, where one objective might be costs to build and another durability of the bridge. For such problems, we need specialized optimizers that determine the Pareto front of mutually non-dominated solutions. There are several established multi-objective evolutionary algorithms (MOEA) and many comparisons on various test functions. However, most of them start with random initial solutions.

If prior knowledge exists or can be generated at a low computational cost, good initial estimates may generate better solutions with faster convergence. These good initial estimates are often referred to as *seeds*, and the method of using good initial estimates is referred to as *seeding*. These botanical terms are used to express the possibility that good solutions for the environment can develop from these starting points. In practice, a good initial seeding can make problem solving approaches competitive that would otherwise be inferior.

For single-objective evolutionary algorithms, methods such as seeding have been studied for about two decades; see, e.g., [17,20,23,26,30,41] for studies and examples (see [27] for a recent categorization). For example, the effects of seeding for the Traveling Salesman Problem (TSP) and the job-shop scheduling problem (JSSP) were investigated in [32]. The algorithms were seeded with known good solutions in the initial population, and it was found that the results were significantly improved on the TSP but not on the JSSP. To investigate the influence of seeding on the optimization, a varying percentage of seeding was used, ranging from 25 to 75%. Interestingly, it was also pointed out that a 100% seed is not necessarily very successful on either problems [28]. This is one of the very few reports that shows seeding can in some cases be beneficial to an optimization process, but not necessarily always is. In [21] a seeding technique for dynamic environments was investigated. There, the population was seeded when a change in the objective landscape arrived, aiming at a faster convergence to the new global optimum. Again, some of the investigated seeding approaches were more successful than others.

One of the very few studies that can be found on seeding techniques for MOEAs is the one performed by Hernandez-Diaz et al. [22]. There, seeds were created using gradient-based information. These were then fed into the algorithm called Non-Dominated Sorting Genetic Algorithm II (NSGA-II, [10]) and the quality was assessed on the benchmark family ZDT ([44], named after the authors Zitzler, Deb, and Thiele). The results indicate that the proposed approach can produce a significant reduction in the computational cost of the approach.

In general, seeding is not well documented for multi-objective problems, even for real-world problems. If seeding is done, then

* Corresponding author. Tel.: +61 8 8313 5405; fax: +61 8 8313 4366.
E-mail address: markus.wagner@adelaide.edu.au (M. Wagner).

http://dx.doi.org/10.1016/j.asoc.2015.04.043
1568-4946/

typically the approach is outlined and used with the comment that it worked in "preliminary experiments"—the reader is left in the dark on the design process behind the used seeding approach. This is quite striking as one expects that humans can construct a few solutions by hand, even if they do not represent the ranges of the objectives well. The least that one should be able to do is to reuse existing designs, and to modify these iteratively towards extremes. Nevertheless, even this manual seeding is rarely reported.

In this paper, we are going to investigate the effects of two structurally different seeding techniques for five algorithms on 48 multi-objective optimization (MOO) problems.

1.1. Seeding

As seeding we use the weighted-sum method, where the trade-off preferences are specified by non-negative weights for each objective. Solutions to these weighted-sums of objectives can be found with an arbitrary classical single-objective evolutionary algorithm. In our experiments we use the algorithm Covariance Matrix Adaptation Evolution Strategy (CMA-ES, [18]). Details of the two studied weighting schemes are presented in Section 2.1.

1.2. Quality measure

There are different ways to measure the quality of the solutions. A recently very popular measure is the hypervolume indicator, which measures the volume of the objective space dominated by the set of solutions relative to a reference point [43]. Its disadvantage is its high computational complexity [4,3] and the arbitrary choice of the reference point. We instead consider the mathematically well founded *approximation constant*. In fact, it is known that the worst-case approximation obtained by optimal hypervolume distributions is asymptotically equivalent to the best worst-case additive approximation constant achievable by all sets of the same size [6]. For a rigorous definition, see Section 2. This notion of multi-objective approximation was introduced by several authors [19,15,31,35,36] in the 80s and its theoretical properties have been extensively studied [9,12,33,34,37].

1.3. Algorithms

We use the jMetal framework [13] and its implementation of NSGA-II [10], Strength Pareto Evolutionary Algorithm (SPEA2, [45]), S-Metric Selection Evolutionary Multi-Objective Algorithm (SMS-EMOA, [14]), and Indicator Based Evolutionary Algorithm (IBEA, [42]). Additionally to these more classical MOEAs, we also study Approximation Guided Evolution (AGE, [7]), which aims at directly minimizing the approximation constant and has shown to perform very well for larger dimensions [38–40]. For each of these algorithms we compare their regular behavior after a certain number of iterations with their performance when initialized with a certain seeding.

1.4. Benchmark families

We compare the aforementioned algorithms on four common families of benchmark functions. These are DTLZ ([11], named after the authors Deb, Thiele, Laumanns and Zitzler), LZ09 ([29], named after the authors Li and Zhang), WFG ([24], named after the authors' research group Walking Fish Group) and ZDT [44]. While the last three families only contain two- and three-dimensional problems, DTLZ can be scaled to an arbitrary number of dimensions.

2. Preliminaries

We consider minimization problems with d objective functions, where $d \geq 2$ holds. Each objective function $f_i : S \mapsto \mathbb{R}$, $1 \leq i \leq d$, maps from the considered search space S into the real values. In order to simplify the presentation we only work with the dominance relation on the objective space and mention that this relation transfers to the corresponding elements of S.

For two points $x = (x_1, \ldots, x_d)$ and $y = (y_1, \ldots, y_d)$, with $x, y \in \mathbb{R}^d$ we define the following dominance relation:

$$x \preceq y \quad :\Leftrightarrow \quad x_i \leq y_i \text{ for all } 1 \leq i \leq d,$$

$$x \prec y \quad :\Leftrightarrow \quad x \preceq y \text{ and } x \neq y.$$

We assess the seeding schemes and algorithms by their achieved *additive approximation* of the (known) Pareto front. We use the following definition.

Definition 1. For finite sets $S, T \subset \mathbb{R}^d$, the additive approximation of T with respect to S is defined as

$$\alpha(S, T) := \max_{s \in S} \min_{t \in T} \max_{1 \leq i \leq d} (s_i - t_i).$$

We measure the approximation constant with respect to the known Pareto front of the test functions. The better an algorithm approximates a Pareto front, the smaller the additive approximation value is. Perfect approximation is achieved if the additive approximation constant becomes 0. However, the approximation constant achievable for a (finite) population with respect to a continuous Pareto front (consisting of an infinite number of points) is always strictly larger than 0. It depends on the fitness function what is the smallest possible approximation constant achievable with a population of bounded size.

2.1. Seeding

For the task of computing the seeds, we employ an evolutionary strategy (ES), because it "self-adapts" the extent to which it perturbs decision variables when generating new solutions based on previous ones. CMA-ES [18] self-adapts the covariance matrix of a multivariate normal distribution. This normal distribution is then used to sample from the multidimensional search space where each variate is a search variable. The co-variance matrix allows the algorithm to respect the correlations between the variables making it a powerful evolutionary search algorithm.

To compute a seed, a (2,4)-CMA-ES minimizes $\sum_{i=1}^{d} a_i f_i(x)$, where the $f_i(x)$ are the objective values of the solution x. In preliminary testing, we noticed that larger population values for CMA-ES tended to result in seeds with better objective values. This came at the cost of significantly increased evaluation budgets, as the learning of the correlations takes longer. Our choice does not necessarily represent the optimal choice across all 48 benchmark functions, however, it is our take on striking a balance between (1) investing evaluations in the seeding and (2) investing evaluations in the regular multi-objective optimization. Note that large computational budgets for the seeding have the potential to put the unseeded approaches at a disadvantage, if the final performance assessment is not done carefully.

The number of seeds, the coefficients used, and the budget of evaluations is determined by the seeding approaches, which we will describe in the following.

CCornersAndCentre: A total of 10,000 evaluations is equally distributed over the generation of $d+1$ seeds. The rest of the