Computers & Operations Research 72 (2016) 50–70

Contents lists available at ScienceDirect

Computers & Operations Research

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

Computational investigation of simple memetic approaches for continuous global optimization

Federico Cabassi, Marco Locatelli *

Dipartimento di Ingegneria dell'Informazione, Università di Parma Via G.P. Usberti, 181/A, 43124 Parma, Italy

ARTICLE INFO

Available online 17 February 2016

Keywords:
Global optimization
Memetic approaches
Funnel landscapes

ABSTRACT

In Locatelli et al. (2014) [20] a memetic approach, called MDE (Memetic Differential Evolution), for the solution of continuous global optimization problems, has been introduced and proved to be quite efficient in spite of its simplicity. In this paper we computationally investigate some variants of MDE. The investigation reveals that the best tested variant of MDE outperforms the original MDE itself, but also that the best variant depends on some properties of the function to be optimized. In particular, a greedy variant of MDE turns out to perform very well over functions with a single-funnel landscape, while another variant, based on a diversity measure applied to the members of the population, works better over functions with a multi-funnel landscape. A hybrid approach is also proposed which combines both the previous variants in order to obtain an overall performance which is good over all functions.

1. Introduction

The task of globally minimizing a multimodal objective function f over some compact domain $X \subset \mathbb{R}^n$ is a very difficult one. Different approaches exist based on the dimension of the search space and on the objective function and feasible domain's properties. The approaches range from exact (usually branch-and-bound) ones, suitable for highly structured problems (e.g., with a quadratic objective function and a polyhedral feasible domain) when the dimension n is not too large (say, few hundreds of variables), to heuristic ones where only few function evaluations are performed, suitable for problems where a single function evaluation is a rather costly operation. While we refer to [19] for a thorough discussion about all possible different approaches, here we focus our attention on approaches which are suitable for problems where local searches are a relatively cheap task but at the same time the huge number of local minimizers rules out the simplest approach based on multiple local searches, namely Multistart, where local searches are performed from different points randomly generated within the feasible region. In the algorithms we are going to discuss the points observed at each iteration are always local minimizers. To be more precise, the observed points are always the output of local search procedures, which are usually guaranteed to be stationary points but, in fact, are typically also local minimizers. In what follows we will always refer to these points as local minimizers, keeping in mind the clarification we have just made. Of course, observing local minimizers has a cost, since for each observation we need to perform a local search, which requires some function (and gradient) evaluations. However, this cost is often largely compensated. Indeed, when the landscape of the objective function is rough with high barriers between local minimizers, even points very close to the global minimizer may have large function values. If, as it is often the case, the function value is employed to evaluate the quality of a point and to decide whether to keep or to discard it, such points, though close to the global minimizer, will be discarded from further consideration. Local searches, by driving a point towards a local minimizer, remove the negative effect of high barriers between local minimizers.

In this paper we discuss three variants of the Memetic Differential Evolution (MDE) approach introduced in [20]. In that paper it was shown that MDE is very competitive, in terms of local searches needed to reach the global minimizer, with respect to other methods based on multiple local searches and, in particular, with respect to the Monotonic Basin Hopping method which will be discussed in Section 2.1. The primary aim of the paper is to show, through an extensive computational investigation, that each proposed variant improves, again in terms of local searches needed to reach the global minimizer, the original MDE approach under some given funnel properties of the objective function to be minimized. More precisely, the greedy variant (G-MDE) improves the performance of MDE when applied to single-funnel

* Corresponding author.
E-mail addresses: federico.cabassi@studenti.unipr.it (F. Cabassi), marco.locatelli@unipr.it (M. Locatelli).

http://dx.doi.org/10.1016/j.cor.2016.01.015
0305-0548/

functions (but with some local search procedures also when applied to multi-funnel functions); the distance variant (D-MDE), by preserving a higher degree of diversity within the population, improves upon the original MDE approach over multi-funnel functions; finally, the hybrid approach (H-MDE), where the greedy and distance strategies are mixed, improves upon the original MDE over all the tested functions. We remark that none of the proposed approaches dominates all the others but this is consistent with the "no free lunch" theorem [43]. As a secondary aim we would like to show that, in spite of its extreme simplicity, one of the proposed variants, namely G-MDE, is competitive with some state-of-the art evolutionary approaches in terms of the quality of the solutions returned within a prefixed budget of function evaluations. The paper is structured as follows. In Section 2 we introduce a general scheme of a global optimization approach based on local searches and we briefly discuss some existing approaches which fit into this scheme, namely Multistart and Mononotic Basin Hopping (Section 2.1), memetic algorithms (Section 2.2), and MDE (Section 2.3) . In Section 3 we propose the three previously mentioned simple variants of MDE. In Section 4 we present the set of test problems on which we compare the different variants and we discuss the results of the computational experiments. We also perform an experimental analysis of the proposed approaches in order to better understand their behavior. Finally, in Section 5 we draw some conclusions and we discuss some possible future developments.

2. Global optimization based on local searches

A general scheme for a global optimization approach based on local searches is displayed in Algorithm 1, where $\mathcal{L}$ is a local solver which input is made up by the objective function *f*, the feasible domain *X* and a starting point $\mathbf{x}$. In this scheme:

Algorithm 1. Generic model for a global optimization algorithm based on local searches.

```
Data: objective function f : ℝ^n → ℝ; feasible domain X; local solver 𝓛(f, X, x)
      where x is the starting point of the local search; parameter vector v;
      population size k.
Result: The best local minimizer x* observed during the execution of the algorithm.
P ← GenerateStartingPoints(f, X, 𝓛, v);
while TerminationCriteria(P, f, X, v) = false do
    for i ∈ {1, ..., k} do
        Q_i ← Generation(f, X, 𝓛, P, v, i);
        P ← Selection(P, Q_i, v, i);
        v ← UpdateParameters(f, X, P, v);
    end
end
```

- $\mathbf{P} \in \mathbb{R}^{n\times k} = \{\mathbf{p}_1, \ldots, \mathbf{p}_k\}$ is the *population matrix* containing the *k* population members as column vectors;
- $\mathbf{Q}_i \in \mathbb{R}^{n\times h} = \{\mathbf{q}_1, \ldots, \mathbf{q}_h\}$ is the *candidate matrix* containing the *h* candidate points generated by the *i*-th member of the current population;
- *GenerateStartingPoints* $(f, X, \mathcal{L}, \mathbf{v})$ is the function returning the initial population;
- *TerminationCriteria* $(\mathbf{P}, f, X, \mathbf{v})$ checks the termination criteria of the algorithm;
- *Generation* $(f, X, \mathcal{L}, \mathbf{P}, \mathbf{v}, i)$ is the function that generates the set of candidate points $\mathbf{Q}_i$;
- *Selection* $(\mathbf{P}, \mathbf{Q}_i, \mathbf{v}, i)$ is the function that performs a selection between the candidate points and the current population;
- *UpdateParameters* $(f, X, \mathbf{P}, \mathbf{v})$ is a function to update the parameter vector.

Algorithm 1 encompasses many different approaches. These can be classified into two broad categories, depending on the cardinality *k* of the population. If $k=1$, then the algorithm is called a *single-track* one, while if $k>1$ the algorithm is a *population-based* one. In what follows we will fix the two functions *GenerateStartingPoints* and *TerminationCriteria*. Both are defined in a rather standard way. The former is described in Algorithm 2 and simply returns *k* local minimizers obtained by running local searches from *k* points randomly generated over *X* ($\mathcal{U}(X)$ is a uniform generator over the set *X*).

Algorithm 2. The function *GenerateStartingPoints* to generate the initial population.

```
Data: f, X and 𝓛.
Result: The matrix P = {p_1, ..., p_k} containing randomly generated local
        minimizers of f over X.
foreach i ∈ {1, ..., k} do
    p_i ← 𝓛(f, X, 𝓤(X));
end
```