Information Sciences 297 (2015) 216–235

Contents lists available at ScienceDirect

Information Sciences

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

Cluster-Based Population Initialization for differential evolution frameworks

Ilpo Poikolainen [a,1], Ferrante Neri [a,b,*], Fabio Caraffini [a,b,1]

[a] Department of Mathematical Information Technology, University of Jyväskylä, P.O. Box 35 (Agora), Jyväskylä 40014, Finland
[b] Centre for Computational Intelligence, School of Computer Science and Informatics, De Montfort University, The Gateway, Leicester LE1 9BH, England, United Kingdom

ARTICLE INFO

Article history:
Received 30 April 2014
Received in revised form 30 October 2014
Accepted 5 November 2014
Available online 15 November 2014

Keywords:
Differential evolution
Memetic computing
Intelligent sampling
Evolutionary computation

ABSTRACT

This article proposes a procedure to perform an intelligent initialization for population-based algorithms. The proposed pre-processing procedure, namely Cluster-Based Population Initialization (CBPI) consists of three consecutive stages. At the first stage, the individuals belonging to a randomly sampled population undergo two subsequent local search algorithms, i.e. a simple local search that performs moves along the axes and Rosenbrock algorithm. At the second stage, the solutions processed by the two local searches undergo the K-means clustering algorithm and are grouped into sets on the basis of their euclidean distance. At the third stage the best individuals belonging to each cluster are saved into the initial population of a generic optimization algorithm. If the population has not been yet filled, the other individuals of the population are sampled within the clusters by using a fitness-based probabilistic criterion. This three stage procedure implicitly performs an initial screening of the problem features in order to roughly estimate the most interesting regions of the decision space. The proposed CBPI has been tested on multiple classical and modern Differential Evolution variants, on a wide array of test problems and dimensionality values as well as on a real-world problem. The proposed intelligent sampling appears to have a significant impact on the algorithmic functioning as it consistently enhances the performance of the algorithms with which it is integrated.

1. Introduction

Engineering and natural sciences often require the solution of multiple optimization problems. This fact makes the study of optimization methods extremely important in fields such as design and control engineering. Since only a very limited number of real-world optimization problems can be solved by exact methods, in the vast majority of cases an optimizer that does not require a specific hypothesis must be used. Over the past decades, computer scientists have designed a multitude of these types of algorithms for addressing real-world problems where an exact approach is almost never applicable. These methods, known as metaheuristics, do not offer guarantees regarding the convergence, but still are capable to detect high

* Corresponding author at: Centre for Computational Intelligence, School of Computer Science and Informatics, De Montfort University, The Gateway, Leicester LE1 9BH, England, United Kingdom. Tel.: +358 14 260 1211; fax +358 14 260 1021.

E-mail addresses: ilpo.poikolainen@jyu.fi (I. Poikolainen), fneri@dmu.ac.uk, ferrante.neri@jyu.fi (F. Neri), fcaraffini@dmu.ac.uk, fabio.caraffini@jyu.fi (F. Caraffini).

[1] Tel.: +358 14 260 1211; fax +358 14 260 1021.

http://dx.doi.org/10.1016/j.ins.2014.11.026
0020-0255/

quality solutions that can be of great interest for engineers and practitioners. Among the plethora of metaheuristics some are Evolutionary Algorithms (EAs) [26], Swarm Intelligence (SI) [25], and Memetic Computing (MC) [50].

For about two decades, i.e. from the 1970s to 1990s, computer scientists have put much effort to design metaheuristics with the intention of detecting an algorithm that could outperform all the other algorithms. After the publication of the No Free Lunch (NFL) Theorems [71], the view on optimization of scientists and practitioners underwent a radical modification. The NFL Theorems prove that all the optimization algorithms, under the hypotheses that they search within a set of finite candidate solutions and that the algorithms never visit the same point/candidate solution twice, display the same performance when averaged over all the possible optimization problems. As an immediate consequence, it was clear that it was no longer useful to discuss which algorithm was universally better or worse. Despite of the fact that the hypotheses of NFL Theorems are often not realistic (for example, it is very unlikely that an EA does not generate the same point twice during a run), a large portion of algorithmic design community started to propose algorithms, which were tailored to specific problems, see e.g. [64,14,53], instead of trying to propose universally applicable algorithms. On the other hand, by using the non-realism of NFL Theorems' hypotheses as an argument, another portion of the optimization community in recent years researchers have attempted to push towards the outer limit of these theorems by proposing relatively flexible algorithmic structures that combine (to some extent) robustness and high performance on various problems. This tendency is especially clear in continuous optimization and for those algorithms characterized by adaptively coordinated heterogeneous algorithmic components. For these two sub-fields, the NFL Theorems are proved to be not verified, see e.g. [3,58], respectively.

Since modern algorithms for continuous optimization are often composed of multiple adaptively coordinated operators, these two sub-fields are not disjointed. For example, in the context of Differential Evolution (DE), the optimizer proposed in [59] combines and coordinates multiple mutation strategies by making use of a learning period and a randomized success based logic (see also [22,52]). In [44] another DE based strategy namely *ensemble* has been presented on the basis of strategy used in Evolutionary Programming proposed in [43]. In the ensemble multiple mutation and crossover strategies, as well as the related parameters are encoded within the solutions and evolve with them. Other harmonic self-adaptive combinations of components within the DE framework are proposed in [8,7]. In the context of Particle Swarm Optimization (PSO), a harmonic coordination of multiple components is also a popular option to enhance the algorithmic robustness over a range of problems. An emblematic example of this strategy is the so-called Frankenstein's PSO [45]. A more elegant algorithm that coordinates, in a simplistic way, a perturbation logic with a variable decomposition is proposed in [37].

Some studies focus on the coordination techniques in order to have a robust behavior of the algorithm. Several nomenclatures are used in different contexts to express fairly similar concepts. With the term *portfolio* it is usually referred to algorithmic frameworks composed of optimizers that are alternatively selected during the run time. The selection criteria can be a simple schedule or a more sophisticated adaptive system. Some examples in the context of continuous optimization are given in [68,57]. In the context of combinatorial optimization, and more specifically for the maximum satisfiability problem, a popular portfolio named SATzilla platform, see [72,32], has been proposed. The difficulty of finding a trade-off between the search algorithms and the aim at determining an automatic coordination system is studied in [33]. A model of the behavior of optimizers in order to predict their run time is presented in [34]. Very closely related to the concept of portfolio, *hyper-heuristics* are composed of multiple algorithms usually coordinated by a machine learning algorithm which takes a supervisory role. This term is in the vast majority of cases, to combinatorial problems. Famous examples of hyper-heuristic have been proposed in [20,11] in the field of timetabling and rostering while in [12] graph coloring heuristics are coupled with a random ordering heuristic. An important concept in hyper-heuristic implementation is the choice function, that is a criterion that assigns a rewarding score to the most promising heuristic, see [20]. More sophisticated coordination schemes present in the literature make use of reinforcement learning in a stand alone or combined fashion, see e.g. [11,23], and memory-based mechanisms, see [10]. Elegant learning schemes coupled with multiple operators (multi-agents) for addressing complex optimization problems are presented in [2,1].

Closely related to hyper-heuristics and portfolio algorithms, Memetic Algorithms (MAs) are optimization algorithms composed of an evolutionary framework and a set of local searchers activated within the generation cycle, see [46,30]. In MAs, as for the related algorithmic families, optimization is carried out by multiple components/sub-algorithms but unlike them, emphasize the global and local search roles of its components. Although the one may argue that there is no clear definition of global and local search (e.g. a DE with a proper tuning can be used as a local search), the term MA is broadly used to refer population-based hybrid algorithms. Moreover, modern MA implementation ignore the original definition that the population-based framework should be evolutionary and refer as MAs also those algorithms based on a SI framework, see e.g. [69,66]. Recently, the concept of MA has been extended to single-solution algorithms or to any algorithm composed of multiple/heterogeneous components. In the latter case the subject is termed, by a part of the computer science community, as Memetic Computing (MC) and its implementation as MC structures, see e.g. [50,49,54,55].

Regardless of the used nomenclature, an important issue, that is also the focus of this paper, is the generation of an initial population in population-based hybrid algorithms. Nearly all the population-based metaheuristics start with the random sampling of a prefixed amount of points within the decision space. This choice can be explained by the motivation, "since we have no a priori knowledge on the problem, we give to each possible candidate solution the same chance to be in the starting population". Obviously, there is nothing wrong in this way of reasoning. Moreover, this initialization has the undoubted advantage that is computationally cheap as it does not require objective function evaluation nor other complex operations. On the other hand, for every problem, there likely exists many other strategies that can lead to much better results. Similar in the motivation, but very different in the implementation, a fully deterministic procedure that spreads