A Variant of Differential Evolution with Enhanced Diversity Maintenance

Joel Chacón Castillo[a], **Carlos Segura**[b]

[a]Computer Science Department
Center for Research in Mathematics (CIMAT)
Guanajuato, México
joelchaconcastillo@gmail.com

[b]Computer Science Department
Center for Research in Mathematics (CIMAT)
Guanajuato, México
carlos.segura@cimat.mx

Evolutionary Algorithms (EAS) are population-based meta-heuristics widely used in complex optimization problems. In spite of their remarkable performance, its behavior can be seriously deteriorated by several reasons. Premature convergence is one of the most important drawbacks that EAS face. A way to alleviate this drawback dwells in the incorporation of diversity management techniques with the aim of attaining a proper balance between exploration and exploitation [1]. In 2013, Črepinšek et al.[2] proposed a quite popular classification of these methods which depends on the sort of components modified in the EA. This taxonomy identifies the following groups: *selection-based*, *population-based*, *crossover/mutation-based*, *fitness-based*, and *replacement-based*. Some of the most successful methods designed in recent years to attain this balance yields in the *replacement-based* group. The basic principle that governs methods belonging to this group is the modification of the level of exploration in successive generations by controlling the diversity of the survivors. In this way, an adequate selection of diverse survivors might slow down the inconvenient of an accelerated convergence. Recent research has shown that important advances are attained when the balance between exploration and intensification is managed by relating the amount of maintained population's diversity to the stopping criterion and elapsed period of execution. Particularly, these methods reduce the importance given to the preservation of diversity as the end of the optimization is approached. This principle has been used to find new best-known solutions for the Frequency Assignment Problem, and to designing the winning strategy of the extended round of Google Hash Code 2020.

In 2019 the "Differential Evolution with Enhanced Diversity Maintenance" (DE-EDM) was proposed, which incorporates a diversity-aware replacement phase to DE. In particular this algorithm explicitly preserves diversity by altering a parameter dynamically. Hence, a dynamic balance between exploration and exploitation is attained with the aim of adapting the optimizer to the requirements of the different optimization stages. DE-EDM was validated with several test problems proposed in competitions of the IEEE Congress on Evolutionary Computation (CEC). In such a comparison, the top-ranked algorithms of each competition (CEC 2016 and CEC 2017), as well other diversity-based schemes were taken into account. The results showed that DE-EDM avoided premature convergence which improved remarkably to state-of-the-art algorithms. Although the benefits of explicitly promoting the diversity in DE are quite evident, those kind of strategies require the setting of two extra user-parameters. Those parameters are the initial distance factor (D_I) and the final moment for diversity promotion (D_F). While the former sets the initial level of diversity required by the replacement operator, the latter is the final moment where penalties based on diversity are performed.

We will present a novel diversity-aware strategy, which is called DE-EDM-II. DE-EDM-II is a simplification of DE-EDM in which the elite vectors are removed, just maintaining a multi-set of target and trial vector. This allows to show that even quite simple variants of diversity-aware DE excel on obtaining really promising

[1]Auto-tuning strategy for evolutionary algorithms: balancing between exploration and exploitation

[2]Exploration and exploitation in evolutionary algorithms: A survey

results. Additionally, we develop a more complete analysis with the aim of better understanding the impact of D_I and D_F on the performance, which shed some light on the reasons for the good performance of these kinds of algorithms in long-term executions.