Paper:

OGDE3: Opposition-Based Third Generalized Differential Evolution

Farid Bourennani, Shahryar Rahnamayan, and Greg F. Naterer

University of Ontario Institute of Technology (UOIT)
2000 Simcoe Street North, Oshawa, Ontario, Canada L1H 7K4
E-mail: farid.bourennani@uoit.ca
[Received October 26, 2011; accepted March 29, 2012]

Multi-Objective Optimization (MOO) metaheuristics are commonly used for solving complex MOO problems characterized by non-convexity, multimodality, mixed-types variables, non-linearity, and other complexities. However, often metaheuristics suffer from slow convergence. Opposition-Based Learning (OBL) has been successfully used in the past for acceleration of single-objective metaheuristics. The most successful example in this regard is Opposition-based Differential Evolution (ODE). However, OBL was not fully explored for MOO metaheuristics. Therefore, in this paper, to the best of our knowledge, for the first time OBL is successfully adapted for a MOO metaheuristic by using a single population (no coevolution). The proposed MOO metaheuristic is based on the GDE3 method and it is called Opposition-based GDE3 (OGDE3). OGDE3 utilizes OBL for opposition-based population initialization and self-adaptive opposition-based generating jumping. Furthermore, the new algorithm is compared with seven state-of-the-art MOO metaheuristics using the ZDT test suite. OGDE3 outperformed the other algorithms; the results are explained and discussed in detail.

Keywords: multi-objective optimization, multi-objective metaheuristic, convergence speed, generalized differential evolution, opposition-based computation

1. Introduction

Most real-world engineering problems require the optimization of multiple conflicting objectives [1]. For example, while designing a power system, electrical engineers are targeting competing objectives such as cost minimization, performance maximization, supply-demand balancing, and so forth. Such problems are called Multi-Objective Optimization (MOO) problems. The solution to a MOO problem is not a single solution. Rather, it is a set of *non-dominated* solutions, called a *Pareto optimal set*. When the real optimal solutions (Pareto-optimal) are plotted in an objective space, they are called a *Pareto front*.

The majority of real-world problems is very complex and cannot be solved by classical optimization methods. Therefore, it is common to use *metaheuristics* which optimize a problem by *iteratively* trying to improve candidate solutions. Especially, MOO metaheuristics are widely used to solve MOO problems characterized by non-differentiability, non-convexity, nonlinearity, multimodality, mixed-types variables, and other types of complexities. However, despite being efficient for many real-world MOO problems, MOO metaheuristics can suffer, depending on the complexity of the problem, from slow convergence speed. They can take over million function calls to find the real front [2]. Hence, MOO metaheuristics convergence speed is an open, unresolved and challenging research problem.

One reason for this slowness is due to metaheuristics generating initially *random* solutions. Then, the metaheuristic tries to converge towards optimal solutions. If random points are generated close to the optimal solutions, it can result in fast convergence. Otherwise, if random guesses are generated far from an existing solution, in a worst case in the opposite location, then the convergence will become much slower or even intractable. Overall, in the absence of any *a priori* knowledge about the problem to optimize, it is not possible to make a good initial guess. Therefore, all directions should be considered simultaneously, or more practically, the opposite direction should be considered, which is called *Opposition-Based Learning* (OBL). Rahnamayan et al. [3] proved mathematically and experimentally that the use of opposite points is more efficient than a pure random exploration.

In this paper, OBL is applied to a state-of-the-art MOO metaheuristic called third Generalized Differentiation Evolution (GDE3) to accelerate its convergence speed. The proposed algorithm, called an Opposition-based third Generalized Differentiation Evolution algorithm (OGDE3), is compared with seven state-of-the-art MOO metaheuristics including its parent algorithm GDE3 against the Zitzler-Deb-Thiele (ZDT) test suite [4]. The results are promising and they are presented and discussed in detail in this paper.

The remainder of the paper is organized as follows. Sections 2 and 3 cover the MOO background and opposition based learning concepts, respectively. Section 4 explains the proposed algorithm, OGDE3. Section 5 presents experimental verifications. Finally, Section 6 concludes the paper.

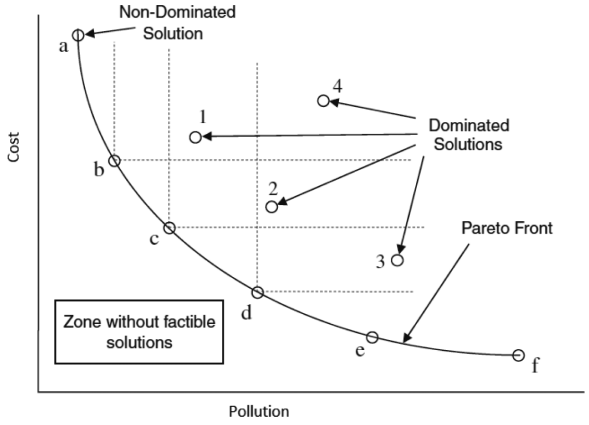

Fig. 1. Pareto front schematic [5].

2. Multi-Objective Optimization: A Background Review

As defined earlier, metaheuristics are computational methods which optimize a problem by *iteratively* trying to improve candidate solutions. In theory, they do not guarantee optimality, but they do provide *feasible* good solutions. Practically, they provide optimal or close to optimal solutions depending on the complexity of the problem.

An MOO problem can be formulated as follows:

$$\text{Min}\, F(x) = [f_1(x), f_2(x), \ldots, f_k(x)]$$

s.t.

$$g_i(x) \geq 0,\ i = 1, 2, \ldots, m$$

$$h_i(x) = 0,\ i = 1, 2, \ldots, p$$

where x is a vector of decision variables for the optimization problem, $f_i(x)$ are objective functions, $g_i(x)$ are unequal constraints, and $h_i(x)$ are equal constraints.

An important aspect of MOO problems is the best trade-off (or compromise) among the different conflicting objectives. By compromising one objective, it is possible to improve another or other objectives. For practical reasons, especially in engineering systems, stakeholders are interested in viewing a variety of these candidate solutions in order to choose the most appropriate solution. As shown in **Fig. 1**, for example, an energy system design might involve two objectives, such as cost and pollution. One solution might be inexpensive but very polluting such as solution f in **Fig. 1**, while another solution can be affordable but more polluting such as solutions c, d, or e. A last solution can be very expensive but almost no pollution such as solution a. The optimal solutions which are *non-dominated* (a, b, c, d, e, f) are called the *Pareto optimal set*. Each solution is presented with a vector, and there are objective functions to calculate the fitness values for each candidate. The plot of the combined objective functions of Pareto optimal sets are collectively known as the *Pareto front*.

There are two main targets when using metaheuristics to solve MOO problems. First, the solutions should converge towards the global optimum. Second, the solutions should be uniformly distributed (diverse) so that the user can have a variety of choices [6]. Ideally, the Pareto set should capture the *complete spectrum* of the Pareto front. However, despite the advantages of metaheuristics, they can be computationally expensive for solving complex problems due to the slow nature of the iterative process. The state-of-the-art algorithms may require thousands to millions function calls [2]. Thus, an efficient multi-objective optimization method should generate accurate and diverse solutions in a timely manner. To accelerate the search space exploration, it is proposed in this paper to use the OBL concept described in the next section.

3. Opposition-Based Learning (OBL)

Opposition concepts have been used for decades in several fields. The following list gives some examples [7].

- Opposite particles/elements (physics)
- Antonyms (language)
- Antithetic variables (simulation)
- Opposite proverbs (culture)
- Complements (set theory)
- Opposition party (politics)
- Inverter (digital design)
- Dualism (philosophy and religion)
- Classical elements (archetype)
- If-then-else (algorithm)
- Complement of an event (probability)
- Revolution (social-politics)

In a similar way, the concept of OBL has been introduced by Tizhoosh [8] for computational intelligence because often a machine learning algorithm starts with initial random points. Then, the algorithm tries to move, hopefully, towards an existing solution. For example, neural network weights, self organising map nodes, and a genetic algorithm population are some of the examples where the initial points are generated randomly. If a random point is generated close to an optimal solution, it can result in fast convergence. However, if a random guess is generated far from an existing solution, in a worst case in the opposite location, then the convergence will become much slower or even intractable. Overall, in the absence of any *a priori* knowledge about the problem to optimize, it is not possible to always make a good initial guess. Therefore, all directions should be considered simultaneously, or more practically, the opposite direction should be considered.