Engineering Applications of Artificial Intelligence 54 (2016) 62–79

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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

Yin-Yang-pair Optimization: A novel lightweight optimization algorithm

Varun Punnathanam, Prakash Kotecha*

Department of Chemical Engineering, Indian Institute of Technology Guwahati, Guwahati 781039, Assam, India

ARTICLE INFO

Article history:
Received 20 November 2015
Received in revised form
18 April 2016
Accepted 19 April 2016
Available online 25 May 2016

Keywords:
Yin-Yang-Pair Optimization
Evolutionary computation
Single objective optimization
Metaheuristic
Congress on Evolutionary Computation 2013

ABSTRACT

In this work, a new metaheuristic, Yin-Yang-Pair Optimization (YYPO), is proposed which is based on maintaining a balance between exploration and exploitation of the search space. It is a low complexity stochastic algorithm which works with two points and generates additional points depending on the number of decision variables in the optimization problem. It has three user defined parameters that provide flexibility to the users to govern its search. The performance of the proposed algorithm is evaluated on the set of problems used for the Single Objective Real Parameter Algorithm competition that was held as part of the Congress on Evolutionary Computation 2013. The results are compared with that of other traditional and recent algorithms such as Artificial Bee Colony, Ant Lion Optimizer, Differential Evolution, Grey Wolf Optimizer, Multidirectional Search, Pattern Search and Particle Swarm Optimization. Based on nonparametric statistical tests, YYPO is shown to provide highly competitive performance relative to the other algorithms while having a significantly lower time complexity. In addition, the performance of YYPO is showcased on three classical constrained engineering problems from literature.

1. Introduction

Metaheuristics are frequently employed to solve optimization problems in various fields due to their inherent advantages over traditional techniques (Deb, 2001). Many techniques such as Genetic Algorithm (GA) (Holland, 1992) and Particle Swarm Optimization (PSO) (Kennedy and Eberhart, 1995) are well established in literature, and variants are regularly developed (Khatib and Fleming, 1998; Ratnaweera et al., 2004; Gülcü and Kodaz, 2015; Tsai, 2015; Kim et al., 2007; Chatterjee and Siarry, 2006; Moayedikia et al., 2015; Idris et al., 2015) to improve or modify certain aspects of the algorithm. This is possible because most metaheuristics are remarkably simple and very flexible. Additionally, being derivative-free, these techniques do not require any prior knowledge on the physics of the problem and hence are attractive options for a wide range of problems (Gosselin et al., 2009; Ramteke and Srinivasan, 2012; Lee et al., 2008; Babu and Angira, 2006; Li and Shao, 2016; Zhao et al., 2016; Ling et al., 2016; Thomas et al., 2015). Single objective metaheuristics also form the basis for more complex algorithms such as multi-objective optimization algorithms. Several popular single objective algorithms have been extended to accommodate multiple objectives in literature (Mirjalili et al., 2016; Deb et al., 2002; Dasheng et al., 2007; Chalermchaiarbha and Ongsakul, 2012; Agrawal et al., 2006).

The No Free Lunch theorem (Wolpert and Macready, 1997) states that no single algorithm can perform well on every optimization problem, encouraging the development of new metaheuristics. These techniques are generally inspired from various everyday phenomena and are predominantly nature inspired. Few such nature inspired algorithms include the Ant-Lion Optimizer (ALO) (Mirjalili, 2015) mimicking the hunting mechanisms of antlions, Artificial Bee Colony (ABC) (Karaboga and Basturk, 2007) based on the foraging behaviour of honey bees, Grey Wolf Optimizer (GWO) (Mirjalili et al., 2014) based on the hierarchy and hunting mechanism of a pack of grey wolves, Differential Evolution (DE) (Storn and Price, 1997) based on the principle of evolution (similar to GA) and Cuckoo Search (Yang and Deb, 2009) simulating the behaviour shown by some cuckoo species in laying their eggs in the nests of other birds. Additionally, various other algorithms that are inspired from miscellaneous phenomena are the Teaching-Learning Based Optimization (Rao et al., 2011) which simulates the classroom environment, the Jaya algorithm (Rao, 2016) which attempts to "approach" the best solution while "moving away" from the worst solution, the Chaotic Golden Section Search Algorithm (Koupaei et al., 2016) based on chaotic maps and the golden section search, the Gravitational Search Algorithm (Rashedi et al., 2009) based on the law of gravity and mass

* Corresponding author.
E-mail address: pkotecha@iitg.ernet.in (P. Kotecha).

http://dx.doi.org/10.1016/j.engappai.2016.04.004
0952-1976/

interactions, Simulated Annealing (SA) (Kirkpatrick et al., 1983) based on the annealing process in metallurgy and Biogeography-based Optimization (Simon, 2008) based on the distribution of biological species over geographies. All the algorithms mentioned above except SA work on a set of solutions (commonly known as population) which communicate among each other to determine an optimal solution. Conversely, SA works on a single solution, and attempts to improve on it over iterations.

The Yin-Yang Algorithm (YYA) (Tam et al., 2011) was developed specifically for the travelling salesman problem and is based on transformation operators from the Chinese classic text *I Ching*. It is a population based technique which utilizes the mutation and selection operators of GA, while employing a transformation operator based on the hexagram transformations of *I Ching* in place of crossover. YYA shows similarities with GA in many levels, but it bears no resemblance with the algorithm proposed in this work. Although both the algorithms have similar names, the similarities end there as the proposed algorithm is not population based and has been developed for bounded real parameter single objective optimization problems as opposed to the travelling salesman problem (although the proposed algorithm may be extended for the same).

Most metaheuristics model a specific phenomenon or mechanism based on which they tackle optimization problems. On the contrary, the Yin-Yang-Pair Optimization (YYPO) algorithm is not based on any specific mechanism or physical event but is designed to explicitly balance exploration and exploitation and thus attempts to be a realization of the Yin Yang philosophy of balance between conflicting forces. Thus, YYPO is the authors' implementation of how the contradicting behaviours inherent in an evolutionary optimization algorithm (exploration and exploitation) can be balanced such that they complement each other to effectively determine the optimal solution. The algorithm is designed for bounded real parameter unconstrained single objective optimization, although the framework may be extended to accommodate constrained as well as mixed integer problems by incorporating minor modifications. For instance, the algorithm can handle constrained as well as mixed integer problems by employing the appropriate techniques from literature (Deb and Agrawal, 1999; Deep et al., 2009).

The rest of this article is structured as follows. The algorithm is described in detail in Section 2 along with a study of its behaviour on a sample problem. Section 3 briefly describes the other optimization algorithms which are used for a comparative performance analysis. Section 4 discusses the benchmark problems along with the test characteristics which are used for studying the performance of YYPO. This performance, including the algorithm complexities, is discussed and compared with that of the other algorithms in Section 5. The performance of YYPO on constrained engineering problems are presented in Section 5 as well. The work is summarized in Section 6 along with a discussion on future directions.

2. Algorithm description

In this section, we discuss the inspiration for the proposed algorithm along with its detailed modelling. Additionally, the behaviour of the algorithm is studied on the two dimensional Rastrigin function for easy visualization.

2.1. Inspiration

Many aspects in the universe are governed by dualities, which refers to two opposite forces or states of conflicting nature being at work. Few examples of mundane dualities are light and darkness, the body and the mind, male and female, good and bad or simply life and death. In the field of science, the wave-particle relationship, positive and negative charges, constants and variables or the binary digits 1 and 0 form typical examples. These dualities are depicted in the Chinese philosophy as Yin and Yang, two complementary and interdependent extremes that would not exist without the other. One aspect gradually changes to the other and this cycle is constantly being repeated, thus the balance between these two aspects results in harmony.

In the field of evolutionary computing, exploitation and exploration represent two conflicting behaviours which work together for solving a problem, and a right balance between them is integral to the performance of the algorithm. The correlation of these behaviours with Yin and Yang is apparent, and this has been the inspiration behind the algorithm.

2.2. Modelling the algorithm

In this algorithm, all the decision variables are handled in their normalized form (between 0 and 1) and are scaled appropriately for functional evaluation with the help of the variable bounds. YYPO employs two points (*P1* and *P2*) among which one of the points (point *P1*) is designed to focus on exploitation, while the other (point *P2*) is designed to focus on exploration of the variable space. These points provide the flexibility to establish a balance between the exploration and exploitation, and this is expected to lead to an ideal performance. The points *P1* and *P2* act as centres to explore the hypersphere volumes in the variable space defined by radii of δ_1 and δ_2 respectively. These radii are self-adaptive such that δ_1 has a tendency to periodically decrease and δ_2 to increase. It should be noted that δ_1 and δ_2 are not user defined parameters and simulate a converging–diverging couple of hyperspheres. The algorithm consists of two main stages, the splitting stage and the archive stage. The splitting stage is encountered at every iteration and is used to explore the hypersphere with the radii (δ) around the two points whereas the archive stage is encountered at dynamic intervals of iterations (I) and updates δ_1 and δ_2 using a user-defined expansion/contraction factor (α). In the following discussion, D is used to denote the problem dimension (number of decision variables of the problem).

The algorithm begins with the generation of two random points in the domain of $[0, 1]^D$ and evaluating their fitness. The fitter of the two points is assigned as *P1* and the other as *P2*. The required parameters in terms of the minimum and maximum number of archive updates (I_{min} and I_{max}) and the expansion/contraction factor (α) are to be specified and the values of δ_1 and δ_2 are set at 0.5. The number of archive updates is randomly generated between (I_{min} and I_{max}). Subsequent to this, the iteration loop is initiated and the fitness of the two points are compared. If *P2* is fitter than *P1*, the points as well as their corresponding δ values are interchanged, ensuring that the iteration starts with the fitter point as *P1*. Both the points are stored in the archive and each point along with its δ enters the splitting stage.

2.2.1. Splitting stage

The inputs to the splitting stage are one of the points (*P1* or *P2*) along with its corresponding search radii (δ_1 and δ_2). Although both points undergo the splitting stage, only a single point (referred as *P*) along with its search radii (δ) undergoes the splitting stage at a time. The splitting stage is designed so as to generate new points in the hypersphere (around the point *P* with the radius δ) at directions as varied as possible, while maintaining a level of randomness. This is implemented by one of the following two methods and is decided based on equal probability.

– **One-way splitting**: In this method, $2D$ identical copies of the