Applied Soft Computing 13 (2013) 2144–2158

Contents lists available at SciVerse ScienceDirect

Applied Soft Computing

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

A novel multi-swarm algorithm for optimization in dynamic environments based on particle swarm optimization

Danial Yazdani[a,*], Babak Nasiri[b], Alireza Sepas-Moghaddam[b], Mohammad Reza Meybodi[c,d]

[a] Young Researchers Club and Elites, Mashhad Branch, Islamic Azad University, Mashhad, Iran
[b] Department of Computer Engineering and Information Technology, Islamic Azad University, Qazvin Branch, Qazvin, Iran
[c] Department of Computer Engineering and Information Technology, Amirkabir University of Technology, Tehran, Iran
[d] Institute for Studies in Theoretical Physics and Mathematics (IPM), School of Computer Science, Tehran, Iran

ARTICLE INFO

Article history:
Received 17 May 2012
Received in revised form 5 September 2012
Accepted 20 December 2012
Available online 3 January 2013

Keywords:
Particle swarm optimization
Dynamic environments
Swarm intelligence
Moving Peak Benchmark
Multi-swarm

ABSTRACT

Optimization in dynamic environment is considered among prominent optimization problems. There are particular challenges for optimization in dynamic environments, so that the designed algorithms must conquer the challenges in order to perform an efficient optimization. In this paper, a novel optimization algorithm in dynamic environments was proposed based on particle swarm optimization approach, in which several mechanisms were employed to face the challenges in this domain. In this algorithm, an improved multi-swarm approach has been used for finding peaks in the problem space and tracking them after an environment change in an appropriate time. Moreover, a novel method based on change in velocity vector and particle positions was proposed to increase the diversity of swarms. For improving the efficiency of the algorithm, a local search based on adaptive exploiter particle around the best found position as well as a novel *awakening–sleeping* mechanism were utilized. The experiments were conducted on Moving Peak Benchmark which is the most well-known benchmark in this domain and results have been compared with those of the state-of-the art methods. The results show the superiority of the proposed method.

1. Introduction

Optimization is considered among the most important problems in mathematics and sciences. The importance of optimization and its numerous applications has inspired the scientists to investigate on different aspects of it. Optimization problems could be seen in real-world applications, e.g. itinerary selection. The goal in all optimization problems is to maximize or minimize one or more cost functions in a problem considering its limitations. While there are a limited number of limitations in a problem space, it can be solved easily. However, increasing limitations leads to an NP-hard problem which needs a high computational cost to be solved. Therefore, researchers are continually seeking the efficient ways for solving such NP-hard problems. Meta-heuristic methods are among these techniques.

Meta-heuristic methods present a computing method for solving optimization problems in which an iterative process for enhancing the obtained solution is utilized until a terminating state is reached. Until now, most existing meta-heuristic methods have focused on static problems. In such problems, the problem space remains unchanged during the optimization process. However, most optimization problems in real world are dynamic and non-deterministic, i.e. the problem search space changes during the optimization process. For example, scheduling tasks is a problem usually solved as a static optimization problem. However, by arriving of a new task during the scheduling procedure, or occurrence of some other problems such as failures in resources, the search environment is changed from a static problem into a dynamic one. As a result, the previous static solutions may no longer be applicable on the new environment. Such problems are called dynamic state optimization problems.

In static optimization problems, finding a global optimum is considered as the main goal. On the other hand, finding a global optimum is not the only goal in dynamic environments and tracking the optimum in the problem space is extremely important in this domain. In fact, the proposed methods for optimization in static environments fail to appropriately follow the optimum. Thus, such methods are not suitable to be used in dynamic environments and the necessity of finding different techniques involving different goal functions and different evaluation criteria for optimization in dynamic environments is obvious.

In this paper, a new optimization method based on PSO has been proposed, by presenting a set of consistency techniques with the problem space for optimization in dynamic environments. To

* Corresponding author. Tel.: +98 935 1185556.
E-mail addresses: danial.yazdani@yahoo.com, d_yazdani@mshdiau.ac.ir (D. Yazdani), nasiri.babak@qiau.ac.ir (B. Nasiri), sepasmoghaddam@qiau.ac.ir (A. Sepas-Moghaddam), mmeybodi@aut.ac.ir (M.R. Meybodi).

1568-4946/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.asoc.2012.12.020

evaluate the proposed method, Moving Peak Benchmark (MPB) has been used, which is the best-known benchmark for evaluating optimization methods in dynamic environments.

The rest of the paper is organized as follows. Section 2 reviews the previous literature on the subject. Section 3 explains the proposed method for solving optimization problems in dynamic environments. Section 4 is dedicated to the experiments and the obtained results. The last section concludes the paper and presents the scope of the future works.

2. Related work

Using meta-heuristic methods for optimization in dynamic environments has its own challenges which do not exist in static environments. The most important challenges encountered by meta-heuristic methods in dynamic environments are *outdated memory* and *diversity loss*. The outdated memory challenge exists in optimization in dynamic environments, because when the environment changes, the fitness value of the obtained solutions will change and will no longer correspond to the stored value in the memory used by these methods.

Diversity loss occurs because of the intrinsic nature of meta-heuristic methods for convergence. The reason for this challenge is the inherent characteristic of these algorithms for converging to the previous optimum position and the exorbitant proximity of the solutions to each other. The easiest way to solve these two issues is re-initialization [1]. In re-initialization, we look at the changed environment as a new problem and re-start the optimization method using the changed environment. However, due to this fact that the efficiency of the optimization process in the changed environment could be improved using the knowledge acquired from the previous environment, the re-initialization methods imply the loss of all the knowledge obtained so far from the problem space.

In the following, the presented solutions for resolving these two main challenges are studied in details. The first challenge, i.e. outdated memory, is of less importance compared to the other one, and two solutions have been proposed for it, which are forgetting memory and re-evaluating memory [2]. These two solutions are also used for optimization methods which use the memory for storing information obtained from the problem space. In forgetting memory method, the position stored for each solution will be replaced by its position in the new environment. In re-evaluating memory method, the stored positions in the memory are re-evaluated.

Several solutions have been proposed for the second challenge, i.e. diversity loss. The solutions have been classified into two main categories: presenting diversity methods and diversity maintenance methods.

2.1. Presenting diversity methods

In this category, the algorithms allow the diversity loss to occur, and afterward they try to solve it. They are divided into two subcategories as well:

2.1.1. Mutation and self-adaptation

In this subcategory, the algorithms try to generate the lost diversity in the environment by performing mutation and self-adaptation. In [3], an adaptive mutation operator called Triggered Hyper-Mutation was proposed as a coefficient which was multiplied by the normal mutation. In [4], a chaotic mutation is used adaptively to create diversity in the environment. Another method is proposed in [5], introducing a variable local search which solves the problem of constancy in mutation step size in [3] by making it adaptive. Replacing the random solutions by the previous suitable solutions after the environment change is another strategy suggested in [6] for producing diversity. In [7] a variable relocation method has been proposed which relocates the solutions according to the amount of change in the fitness function value at the time of the environment change, performing such relocations for each solution with different radiuses. Also, in [55] a hyper-reflection scheme is presented for optimization in dyanmic environments.

2.1.2. Other approaches

There are other methods proposed for creating diversity in the environment after discovering a change in the environment. In [8], a method called RPSO has been proposed for randomizing some or all of the solutions after detecting changes in the environment. In [9], another algorithm called PBIL (Population-Based Incremental Learning) is presented which uses a flexible probability vector for generating solutions. In [56] a hybrid approach for dynamic continuous optimization problems is presented and in [57] a cooperative meta-heuristic approach configured via fuzzy logic and SVMs.

2.2. Diversity maintenance methods

In this method, it is tried to keep diversity in the environment at all times (before and after the change). The proposed algorithms in this category are divided into three subcategories:

2.2.1. Dynamic topology

Algorithms in this subcategory try to decrease the rate of algorithm convergence into a global optimum by limiting the existing communications among solutions, and thus keep the diversity in the environment. In [10], a method called FGPSO with a proximity structure similar to a grid is proposed for maintaining diversity. In [11], HPSO has been proposed for preserving diversity, with its hierarchical and pseudo-tree structure. Another method, called Cellular PSO [12] has been suggested for optimization in dynamic environments, and this method utilizes the local information exchange and cellular automata distribution features. In [13], the presented model in [12] has been improved by changing the roles of some particles to quantum particles just after the environment change.

2.2.2. Memory-based methods

When the environment changes are periodical or recurrent, it will be very useful to store previous optimal solutions for using in the future. Memory-based methods try to store such information. These methods have been usually suggested for evolutionary methods such as GA and EDA which are genetic-based. Memory-based methods themselves can be divided into two main subclasses of implicit memory and explicit memory.

2.2.2.1. Implicit memory. In this subcategory of memory-based methods, memory is integrated with meta-heuristic methods as a redundant representation. Redundant representation using diploid genomes is the most widely used method in this subclass [14,15]. In diploid genome, each solution consists of two alleles in every locus. In [14,15], two methods based on diploid genetic algorithms have been proposed for optimization in dynamic environments using implicit memory methods.

2.2.2.2. Explicit memory. In this subclass, memory is created explicitly. This subclass has its own two subclasses as follows:

(a) *Direct memory*: most of the times, direct memories include the previous suitable solutions [16], but in some cases solutions with the most diversity are also kept in memory.
(b) *Associative memory*: a wide variety of information is stored in associative memories, such as the maintained information about the environment in certain times, a list of environment variables and the probability of their state change [17], the probability of occurrence of a suitable solution in a certain