Neurocomputing 172 (2016) 356–370

Contents lists available at ScienceDirect

Neurocomputing

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

History-Driven Particle Swarm Optimization in dynamic and uncertain environments

Babak Nasiri [a,*], MohammadReza Meybodi [b], MohammadMehdi Ebadzadeh [b]

[a] Department of Computer Engineering and Information Technology, Qazvin Branch, Islamic Azad University, Qazvin, Iran
[b] Department of Computer Engineering and Information Technology, Amirkabir University of Technology, Tehran, Iran

ARTICLE INFO

Article history:
Received 21 August 2014
Received in revised form
7 May 2015
Accepted 12 May 2015
Available online 5 August 2015

Keywords:
Dynamic optimization
Particle swarm optimization
History-Driven approach
Dynamic environments
Swarm intelligence

ABSTRACT

Due to dynamic and uncertain nature of many optimization problems in real-world, an algorithm for applying to this environment must be able to track the changing optima over the time continuously. In this paper, we report a novel multi-population particle swarm optimization, which improved its performance by employing an external memory. This algorithm, namely History-Driven Particle Swarm Optimization (HdPSO), uses a BSP tree to store the important information about the landscape during the optimization process. Utilizing this memory, the algorithm can approximate the fitness landscape before actual fitness evaluation for some unsuitable solutions. Furthermore, some new mechanisms are introduced for exclusion and change discovery, which are two of the most important mechanisms for each multi-population optimization algorithm in dynamic environments. The performance of the proposed approach is evaluated on Moving Peaks Benchmark (MPB) and a modified version of it, called MPB with pendulum motion (PMPB). The experimental results and statistical test prove that HdPSO outperforms most of the algorithms in both benchmarks and in different scenarios.

1. Introduction

Over the past decade, there has been a growing interest in applying swarm intelligence and evolutionary algorithm for optimization in dynamic and uncertain environments [1]. The most important reason for this growing can be related to the dynamic nature of many optimization problems in real-world. World Wide Web (WWW) is an outstanding example of such an environment with multiple dynamic problems. Web page clustering, web personalization and web community identification are some of the most well-known sample problems from this dynamic environment.

Generally, the goals, challenges, performance measures and the benchmarks are completely different from the static to dynamic environments. In static optimization problems, the goal is only finding the global optima. However, in dynamic optimization problems, detecting change and tracking global optima as closely as possible are the other goals for optimization. Also, in many cases in real-world optimization problems, the old and new environments are somehow correlated to each other. Therefore, a good optimization algorithm also must be able to learn from the previous environments to work better [2].

From the challenges exist for optimization in static environments, we can mention to premature convergence, trapping in local optima and trade-off between exploration and exploitation. Although, optimization in dynamic environments has some new challenges such as Detecting change in environment, transforming a local optima to a global optima and vice versa, losing diversity after change, outdated memory after change, chasing one peak by more than one sub-population in multi-modal environment, unknown severity of change in environment, changing part of environment instead of the entire environment, unknown time dependency between changes in environment.

Up to now, multiple benchmarks and performance measures are defined in literature for evaluation the optimization algorithms in dynamic environments, which are completely different from static ones. Offline error, Offline performance [2] and best-before-change error [3] are some of the most commonly used measures in literature. Furthermore, MPB [4], DF1 [5], XOR dynamic problem generator [6]

* Corresponding author. Tel.: +98 9123442922.
E-mail address: Nasiri.babak@qiau.ac.ir (B. Nasiri).

http://dx.doi.org/10.1016/j.neucom.2015.05.115
0925-2312/© 2015 Elsevier B.V. All rights reserved.

are some of the most commonly used benchmarks in state of the art. In this paper, we uses the MPB and a modified version of it, called PMPB as the benchmark problems and offline error as the performance measure for evaluation of the proposed algorithm.

Over the past two decades, many optimization algorithms are proposed based on swarm intelligence and evolutionary algorithm for multimodal, time-varying and dynamic environments. The most widely used algorithms are PSO [7–13], GA [6,14–16], ACO [17–19], DE [20–30] and ABC [31–36].

We can divide the proposed approaches for optimization in dynamic environments into six different groups: (1) introducing diversity when changes occur, (2) maintaining diversity during the change, (3) memory-based approaches, (4) prediction approaches, (5) self-adaptive approaches and (6) multi-population approaches. In this paper, we use a combination of second, third and sixth approaches to take advantages of the whole of them.

In many dynamic optimization problems in real-world, the current state is often similar or related to the previously seen states. Using the past information may help the algorithm to be more adaptive to the changes in the environment and to perform better over time. One way to maintain the past information is the use of memory in implicit or explicit type. Implicit memory stores the past information as part of an individual, whereas explicit one stores information separate from the population. Explicit memory has been much more widely studied and has been produced much better performance on dynamic optimization problem than implicit one.

Explicit memory can be divided into direct and associative memory. In most cases, the information in direct memories are the previous good solutions [37,38]. However, associative memory can be included various type of information such as the probability vector that created the best solutions [37], the probability of the occurrence of good solutions in the landscape [39].

In [40,41] a non-revisiting genetic algorithm (NrGA) is proposed, which memorize all the solutions by a binary space partitioning (BSP) tree structure. This scheme uses the BSP tree as an explicit memory in a static optimization problem to prevent the solutions evaluates more than one time during the run. Also, [42] utilized the BSP tree in continuous search space to guide the search by doing some adaptation on the mutation operator. In [43,44], the BSP tree is utilized for creating an adaptive parameter control system. It automatically adjusts the parameters based on entire search historical information in a parameter-less manner. In this paper, the authors used the BSP tree for different aims, which include improving the exploration capability of the finder swarm, improving the exploitation capability of fine-tuning mechanism and proposing different mechanism for exclusion.

Unlike many other direct memory approaches, the proposed approach stores all the past individuals evaluated during each environment. At the first glance, it seems that it needs a large amount of memory and it is not an economical method. However, in many optimization problems in real-world, the fitness evaluation cost is very higher than individual generation cost. For such problems, the total number of fitness evaluation is limited and it is a mistake to throw away any information achieved from the past fitness evaluation [42].

In this paper, we report a novel swarm intelligence algorithm, namely History-Driven Particle Swarm Optimization (HdPSO) for optimization in dynamic environments. It is shown that using explicit memory during the run can help the algorithm to guide the local search and global search toward the promising area in the search space and remember the previous environments if they happen again. Moreover, we incorporate the BSP tree to approximate the fitness for the individuals before actual fitness evaluation and in this way, we decrease the number of wasted fitness evaluation for some inappropriate individuals and improve the performance of the algorithm significantly.

It is worth mentioning that the proposed approach can be seen as a framework for applying optimization algorithms on dynamic environments. Most of the techniques applied on standard particle swarm optimization in this paper, can be also applied on more powerful optimization algorithms such as CoBiDE [45], CoDE [46], BSA [47] and DSA [48] which are introduced recently.

The remainder of this paper is organized as follows: Section 2 presents the fundamental of the proposed HdPSO, and the structure of the explicit memories used in this work in detail. Section 3 reports the experimental results on moving peaks benchmark and the new modified version of it with various configurations. Finally, Section 4 presents some concluding remarks.

2. Proposed algorithm

In the following subsections, we first describe how an explicit memory-based approach work generally. Then, we describe the overall scheme of the proposed HdPSO algorithm in detail.

2.1. Explicit memory-based approach

In this paper, an explicit memory-based approach is proposed for storing the past useful information during the computation. This explicit memory has two parts, including long-term and short-term memories.

The long-term memory is responsible for storing the most important information about the past environments. This information includes the position and fitness value of the all discovered local and global optima in the past environments. Taking advantage of such a memory, an algorithm has the ability to remember the previous environments, if they re-appeared in the future exactly or with some noise. The information in the long-term memory is persistent and does not disappear by the happening changes in environment. However, it becomes more complete and more powerful by happening each change in environment. Fig. 1 shows the structure of this long-term memory in more detail.

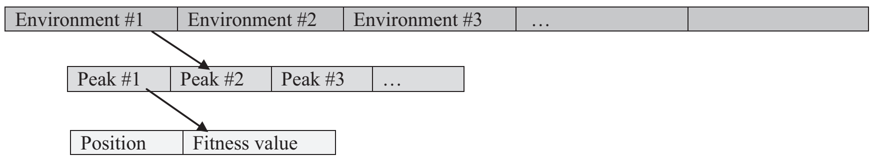

Fig. 1. The structure of the long-term memory in detail.