Applied Soft Computing 32 (2015) 432–448

Contents lists available at ScienceDirect

Applied Soft Computing

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

A novel hybrid adaptive collaborative approach based on particle swarm optimization and local search for dynamic optimization problems

Ali Sharifi [a], Javidan Kazemi Kordestani [b,*], Mahshid Mahdaviani [a], Mohammad Reza Meybodi [a]

[a] Soft Computing Laboratory, Computer Engineering and Information Technology Department, Amirkabir University of Technology (Tehran Polytechnic), 424 Hafez Ave., Tehran, Iran
[b] Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran

ARTICLE INFO

Article history:
Received 6 August 2014
Received in revised form 1 January 2015
Accepted 2 April 2015
Available online 13 April 2015

Keywords:
Dynamic optimization problems
Moving peaks benchmark
DOPs
MPB
Particle swarm optimizer
Naive direct search

ABSTRACT

This paper proposes a novel hybrid approach based on particle swarm optimization and local search, named PSOLS, for dynamic optimization problems. In the proposed approach, a swarm of particles with fuzzy social-only model is frequently applied to estimate the location of the peaks in the problem landscape. Upon convergence of the swarm to previously undetected positions in the search space, a local search agent (LSA) is created to exploit the respective region. Moreover, a density control mechanism is introduced to prevent too many LSAs crowding in the search space. Three adaptations to the basic approach are then proposed to manage the function evaluations in the way that are mostly allocated to the most promising areas of the search space. The first adapted algorithm, called HPSOLS, is aimed at improving PSOLS by stopping the local search in LSAs that are not contributing much to the search process. The second adapted, algorithm called CPSOLS, is a competitive algorithm which allocates extra function evaluations to the best performing LSA. The third adapted algorithm, called CHPSOLS, combines the fundamental ideas of HPSOLS and CPSOLS in a single algorithm. An extensive set of experiments is conducted on a variety of dynamic environments, generated by the moving peaks benchmark, to evaluate the performance of the proposed approach. Results are also compared with those of other state-of-the-art algorithms from the literature. The experimental results indicate the superiority of the proposed approach.

1. Introduction

Optimization in dynamic environments has emerged as an important field of research during the last two decades, since many real-world optimization problems tend to change over time. The representative examples of real-world dynamic optimization problems (DOPs) are portfolio decisions optimization in changing stock market conditions, shortest path routing problem in changing network environments, and vehicle routing with online arrival of customer's requests. In these problems, the task of optimization is not confined to finding the optimum solution(s) in the problem space but to continuously and accurately adapt to the new conditions during the course of optimization.

Each dynamic optimization problem P can be defined by a quintuple $\{\Omega, x, \phi, f, t\}$ where Ω denotes the search space, x is a feasible solution in Ω, ϕ represents the system control parameters which determines the distribution of the solutions in the fitness landscape, f is the static objective function and t is the time. P is then can be modeled as follows [1]:

$$P = \sum_{t=0}^{\text{end}} f_t(x, \phi) \tag{1}$$

As can be seen in the above equation, dynamic optimization problem P is composed of a sequence of static instances. Hence, the goal of the optimization in such problems is no longer just locating the optimal solution(s), but rather tracking the shifting optima over the time.

* Corresponding author. Tel.: +98 9373908326.
E-mail addresses: en.sharifi.ali@gmail.com (A. Sharifi), javidan.kazemi@gmail.com (J. Kazemi Kordestani), mahshid.mahdaviani@gmail.com (M. Mahdaviani), mmeybodi@aut.ac.ir (M.R. Meybodi).

http://dx.doi.org/10.1016/j.asoc.2015.04.001
1568-4946/

The dynamism of the problem can then be obtained by tuning the system control parameters as follows:

$$\phi_{t+1} = \phi_t \oplus \Delta\phi \tag{2}$$

where $\Delta\phi$ is the deviation of the control parameters from their current values and $\oplus$ represents the way the parameters are changed. The next state of the environment then can be defined using current state of the environment as follows:

$$f_{t+1}(x, \phi) = f_t(x, \phi_t \oplus \Delta\phi) \tag{3}$$

Different change types can be defined, using $\Delta\phi$ and $\oplus$. Li et al. [1] proposed a framework of the eight change types including *small step change*, *large step change*, *random change*, *chaotic change*, *recurrent change*, *recurrent change with noise*, *and dimensional change*. For additional information, interested readers are referred to [1].

The adaptive nature of evolutionary algorithms and swarm intelligence methods make them suitable candidates for dealing with DOPs. However, the dynamic behavior of DOPs poses additional challenges to the evolutionary algorithms and swarm intelligence techniques. The main challenge of the mentioned techniques in dynamic environments is *diversity loss* which arises due to the tendency of the population to converge to a single optimum. Consequently, when a change occurs in environment, the number of function evaluations required for a partially converged population to re-diversify and re-converge to the shifted optimum is quite deleterious to the performance [2]. Particle swarm optimization (PSO) is one of the swarm intelligence algorithms that should undergo a certain adjustment to work well in dynamic environments.

In this paper, a novel PSO-based approach which combines a *fuzzy social-only model* PSO and local search is suggested to address DOPs. The basic algorithm is then further extended based on three resource management schemes. Experimental studies are carried out on the sensitivity analysis of the several components and parameters of the proposed method. The performance of the proposed approach is also compared with several state-of-the-art algorithms on moving peaks benchmark (MPB) problem.

The rest of this paper is structured as follows: Section 2 gives an overview to the basic PSO algorithm and its variants for dynamic environments. In Section 3, we explain our proposed approach in detail. Experimental study regarding the parameters sensitivity analysis, impact of using different local search methods, effect of applying different resource management schemes, and comparison with other state-of-the-art methods is presented in Section 4. Finally, conclusions and future works are given in Section 5.

2. Background and related works

PSO is a versatile population-based stochastic optimization method which was first proposed by Kennedy and Eberhart [3] in 1995. PSO begins with a population of randomly generated particles in a *D*-dimensional search space. Each particle *i* of the swarm has three features: $\vec{x}_i$ that shows the current position of the particle *i* in the search space, $\vec{v}_i$ which is the velocity of the particle *i* and $\vec{p}_i$ which denotes the best position found so far by the particle *i*. Each particle *i* updates its position in the search space, at every time step *t*, according to the following equations:

$$\vec{v}_i(t+1) = w\vec{v}_i(t) + c_1 r_1[\vec{p}_i(t) - \vec{x}_i(t)] + c_2 r_2[\vec{p}_g(t) - \vec{x}_i(t)] \tag{4}$$

$$\vec{x}_i(t+1) = \vec{x}_i(t) + \vec{v}_i(t+1) \tag{5}$$

where w is the inertia weight parameter which determines the portion of velocity preserved during the last time step of the algorithm. c_1 and c_2 denote the cognitive and social learning factors that are used to adjust the degree of the movement of particles toward their personal best position and global best position of the swarm, respectively. r_1 and r_2 are two independent random variables drawn with uniform probability from [0,1]. Finally, $\vec{p}_g$ is the globally best position found so far by the swarm. The pseudo-code of the PSO is shown in Algorithm 1.

Algorithm 1. Pseudo-code for canonical PSO

```
Setting parameters
Generate the initial swarm of the particles with random positions and velocities
Evaluate the fitness of each particle of the swarm
repeat
  for each particle i in the swarm do
    update particle i according to Eqs.(4),(5);
    if(fitness(p_i)<fitness(x_i)) then
      p_i:= x_i;
      if(fitness(p_g)<fitness(x_i)) then
        p_g:= x_i;
      end-if
    end-if
  end-for
until a termination condition is met
```

PSO has been successfully applied in many optimization problems including numerical optimization [4,5], image processing [6], training feedforward neural networks in dynamic environments [7], density estimation [8], multi-objective optimization [9,10], data clustering [11], etc. However, as we mentioned in Section 1, PSO suffers from *diversity loss* when dealing with DOPs. Many attempts have been made by the researchers to deal with this issue. In the rest of this section, we will examine the current studies and advances in the literature in five major groups.

2.1. Injecting diversity to the swarm after detecting a change in the environment

A very basic strategy to cope with *diversity loss* problem is to inject a certain level of diversity to the swarm whenever a change occurs in the environment. An example of this approach is the re-randomization PSO (RPSO) proposed by Hu and Eberhart [12]. In RPSO, upon detecting a change in the environment, the entire or a part of the swarm is randomly relocated in the search space. Their results suggest that the re-randomization of a small portion of the