Information Sciences 291 (2015) 19–42

Contents lists available at ScienceDirect

Information Sciences

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

Opposition-based learning in shuffled frog leaping: An application for parameter identification

Morteza Alinia Ahandani [*], Hosein Alavi-Rad

Department of Electrical Engineering, Langaroud Branch, Islamic Azad University, Langaroud, Iran
Young Researchers Club, Langaroud Branch, Islamic Azad University, Langaroud, Iran

ARTICLE INFO

Article history:
Received 10 September 2013
Received in revised form 13 May 2014
Accepted 12 August 2014
Available online 28 August 2014

Keywords:
Opposition-based learning
Shuffled frog leaping
Memeplex
Parameter identification

ABSTRACT

This paper proposes using the opposition-based learning (OBL) strategy in the shuffled frog leaping (SFL). The SFL divides a population into several memeplexes and then improves each memeplex in an evolutionary process. The OBL by comparing the fitness of an individual to its opposite and retaining the fitter one in the population accelerates search process. The objective of this paper is to introduce new versions of the SFL which employ on one hand the OBL to accelerate the SFL without making premature convergence and on the other hand use the OBL strategy to diversify search moves of SFL. Four versions of SFL algorithm are proposed by incorporating the OBL and the SFL. All algorithms similarly use the opposition-based population initialization to achieve fitter initial individuals and their difference is in applying opposition-based generation jumping. Experiments are performed on parameter identification problems. The obtained results demonstrate that incorporating the opposition-based strategy and SFL performed in a proper way is a good idea to enhance performance of SFL. Two versions of opposition-based SFL outperform their pure competitor i.e., SFL in terms of all aspects on all problems but two other versions of SFL obtained a worse performance than the pure SFL. Also some performance comparisons of the proposed algorithms with some other algorithms reported in the literature confirm a significantly better performance of our proposed algorithms. Also in final part of the comparison study, a comparison of the proposed algorithm in this study in respect to other algorithms on CEC05 functions demonstrates a completely comparable performance of proposed algorithm.

1. Introduction

The system identification has been a classical problem in control engineering. It is performed in four stages: (i) Data acquisition. It is the process of sampling signals that measure real world physical phenomenon such as voltage, current, temperature, pressure, or sound and, in the modern applications, converting the resulting samples into digital numeric values that can be manipulated by a computer. (ii) Model structure selection. The choice of a model structure is based upon an understanding of both the identification procedure, and the system to be identified or in the other words understanding of the physical systems performed using three most commonly types of models i.e. the black-box model, gray-box model,

[*] Corresponding author at: Department of Electrical Engineering, Langaroud Branch, Islamic Azad University, Langaroud, Iran. Tel.: +98 142 5244413, mobile: +98 911 9465164; fax: +98 142 5244413.
E-mail addresses: alinia@iaul.ac.ir (M.A. Ahandani), alavi@iaul.ac.ir (H. Alavi-Rad).

http://dx.doi.org/10.1016/j.ins.2014.08.031
0020-0255/© 2014 Elsevier Inc. All rights reserved.

```
Algorithm 1 (the SFL algorithm)
Begin SFL
Step 1: generate and evaluate initial population of size $N_{pop}$.
Step 2: generate $m$ memplexes with $n$ members where $N_{pop} = m \times n$.
Step 3: apply the evolutionary process (Step3. 0 to Step 3.8) to improve each memplex for $k_{\max}$ iterations:
    Step 3.0: set counter $k = 1$.
    Step 3.1: While $k \le k_{\max}$
    Step 3.2: specify the best frog in entire population ( $x_g$ ).
    Step 3.3: specify the worst and the best frogs in the current memeplex ( $x_w, x_b$ )
    Step 3.4: generate and evaluate the new frog ( $x_n$ ) according to Eqs. (1) and(2).
    Step 3.5: if $f(x_n) < f(x_w)$ then $x_w = x_n$ and go to Step 3.8.
    Step 3.6: replace $x_b$ with $x_g$ and repeat Step 3.4 and Step 3.5.
    Step 3.7: generate a random point and replace with $x_w$.
    Step 3.8: set $k = k + 1$.
    End While
Step 4: shuffle the population.
Step 5: check the stopping criteria if are not met go to Step 2.
End SFL
```

Fig. 1. The steps of SFL.

```
Algorithm 2 (the SOBFL algorithm)
Begin SOBFL
Step 1: generate and evaluate initial population of size $N_{pop}$.
Step 2: generate $m$ memplexes with $n$ members where $N_{pop} = m \times n$.
Step 3: apply the evolutionary process (Step 3.0 to Step 3.9) to improve each memplex for $k_{\max}$ iterations:
    Step 3.0: set counter $k = 1$.
    Step 3.1: While $k \le k_{\max}$
    Step 3.2: specify the best frog in entire population ( $x_g$ ).
    Step 3.3: specify the worst and the best frogs in the current memeplex ( $x_w, x_b$ )
    Step 3.4: generate and evaluate the new frog ( $x_n$ ) according to Eqs. (1) and(2).
    Step 3.5: if $f(x_n) < f(x_w)$ then $x_w = x_n$ and go to Step 3.8.
    Step 3.6: replace $x_b$ with $x_g$ and repeat Step 3.4 and Step 3.5.
    Step 3.7: generate a random point and replace with $x_w$.
    Step 3.8: apply "opposition-based generation jumping" strategy based on a jumping rate of $J_r$ :
            If $rand(0,1) < J_r$
            For $i = 1 : n$
                For $j = 1 : D$
                    $Omem_{i,j} = MINmem_j^p + MAXmem_j^p - mem_{i,j}$;
                End For
            End For
            Select $n$ fittest members from the set of $\{mem \cup Omem\}$ as current memeplex.
    Step 3.9: set $k = k + 1$.
    End While
Step 4: shuffle the population.
Step 5: check the stopping criteria if are not met go to Step 2.
End SOBFL
```

Fig. 2. The steps of SOBFL.

and user-defined model. (iii) Parameter estimation. Value of some parameters related to be chose model affects the distribution of the measured data. An estimator attempts to approximate the unknown parameters using the measurements. (iv) Model validity tests. This stage state that how well the model represents the patterns seen in real-world applications and