Information Sciences 334–335 (2016) 44–64

Contents lists available at ScienceDirect

Information Sciences

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

Biasing the transition of Bayesian optimization algorithm between Markov chain states in dynamic environments

Marjan Kaedi[a], Nasser Ghasem-Aghaee[a], Chang Wook Ahn[b,*]

[a] Faculty of Computer Engineering, University of Isfahan, Hezar-Jerib St., Isfahan 81746-73441, Iran
[b] Department of Computer Engineering, Sungkyunkwan University, 2066 Seobu-ro, Suwon 440-746, Republic of Korea

ARTICLE INFO

Article history:
Received 27 May 2013
Revised 10 September 2015
Accepted 10 November 2015
Available online 26 November 2015

Keywords:
Bayesian optimization algorithm
Dynamic environments
Markov chain
Memory-based method

ABSTRACT

When memory-based evolutionary algorithms are applied in dynamic environments, the certainly use of uncertain prior knowledge for future environments may mislead the evolutionary algorithms. To address this problem, this paper presents a new, memory-based evolutionary approach for applying the Bayesian optimization algorithm (BOA) in dynamic environments. Our proposed method, unlike existing memory-based methods, uses the knowledge of former environments probabilistically in future environments. For this purpose, the run of BOA is modeled as the movements in a Markov chain, in which the states become the Bayesian networks that are learned in every generation. When the environment changes, a stationary distribution of the Markov chain is defined on the basis of the retrieved prior knowledge. Then, the transition probabilities of BOA in the Markov chain are modified (biased) to comply with the defined stationary distribution. To this end, we employ the Metropolis algorithm and modify the K2 algorithm for learning the Bayesian network in BOA in order to reflect the obtained transition probabilities. Experimental results show that the proposed method achieves improved performance compared to conventional methods, especially in random environments.

1. Introduction

For many real-world problems, their environments might change during the optimization process. Environmental changes can take various forms such as changes in the parameters, objective functions, or constraints [3,31,50]. When the environment varies, the optimum of that problem tends to be altered as well. Dynamic optimization problems may take into account any of these uncertain events in the optimization process [18,30,46]. Examples include scheduling problems where new jobs arrive over time [4,43], vehicle routing problems where new request arrivals vary with time [4], portfolio-decision optimization problems where the stock market changes and the decisions should be updated consistently [3], and flight path optimization problems where the flight paths should be adapted to special events [3]. In the dynamic optimization problems, the objective is not only finding the optimum solution but also tracking it in a timely manner [51]. In cases where the problem after the environmental change remains similar to previous problems, we can use the knowledge acquired while searching for previous solutions to find the solution to the new problem [18]. Thus, searching for a new solution based on the previous solutions can save computation time [45].

Many real-world dynamic problems have been solved by various soft computing techniques, such as neural networks, particle swarm optimization, and ant colony algorithm [5,6,24,26,33,44,47,54]. Evolutionary algorithms inspired by natural evolution

* Corresponding author. Tel.: +82312994588; fax: +82312994664.
E-mail addresses: Kaedi@eng.ui.ac.ir (M. Kaedi), aghaee@eng.ui.ac.ir (N. Ghasem-Aghaee), cwan@skku.edu, cwan@evolution.re.kr (C.W. Ahn).

http://dx.doi.org/10.1016/j.ins.2015.11.030
0020-0255/© 2015 Elsevier Inc. All rights reserved.

can be especially powerful tools for solving dynamic optimization problems [51]. As a class of evolutionary algorithms [10], estimation of distribution algorithms (EDAs) [22,36,37] have been proposed to resolve the “building blocks destruction” problem. In EDAs, instead of using the standard (genetic) operators, probabilistic models that describe high-quality solutions are used to generate new solutions [29]. Various versions of EDAs differ in their probabilistic models. Meanwhile, numerous evolutionary methods have been proposed to handle the dynamic optimization problems. These methods are classified as follows [2]: (a) reaction of the algorithm after the change, (b) preserving the diversity during the run, (c) memory-based approach, and (d) multi-population approach. In the memory-based approach, evolutionary algorithms are supported by a memory that stores useful information regarding the optimization problems in the previous environments. Compared to other evolutionary algorithms, EDAs have more potential to be used as memory-based dynamic optimization approaches because their probabilistic models contain abstract information about the previous environments [38]. These models can be efficiently stored in the memory, and this memory can be managed in a simple, rapid manner [38]. When environmental change occurs, the relevant information is retrieved and used to solve the new problem [18].

As mentioned, EDAs have greater potential than other evolutionary algorithms to be used for memory-based dynamic optimization approach, and some researchers have successfully applied EDAs to memory-based dynamic optimization. Section 2 reviews these studies.

There is a common feature in all EDA approaches: knowledge of the previous environments is stored in the memory; when a change occurs, the relevant knowledge is retrieved and utilized directly to solve the new problem. When the environmental changes are repetitive or the new environments are similar to the former ones, the retrieved knowledge can be compatible with the new environments. Hence, memory-based methods that deterministically utilize former knowledge are highly efficient. When a change occurs in new environments that are not sufficiently similar to the former ones, no compatible knowledge will be found in the memory. In addition, the similarity measure used for searching the memory and finding the relevant knowledge might not be reliable. In that case, knowledge that seems compatible in the new environment would mislead evolutionary algorithms, and making use of the retrieved knowledge in a deterministic manner would not be effective. The performance achieved with this knowledge may even be lower than when no prior knowledge is used because this inaccurate knowledge can bias evolutionary algorithms toward an incorrectly estimated region in the search space, resulting in the algorithms becoming stuck at sub-optimal points.

In this paper, we address the problem of memory-based evolutionary algorithm misguidance from use of inaccurate prior knowledge. We study the probabilistic use of uncertain prior knowledge in dynamic environments and develop, using Markov chain modeling, a new memory-based method that employs the BOA. To this end, the run of BOA in the dynamic environments is modeled as the movements between the states in a heterogeneous Markov chain. When the environment changes, relevant knowledge is retrieved from the memory and it is used to define a stationary distribution (which persists forever once it is reached) for visiting the states of the Markov chain. Then the transitions of BOA between the states are controlled in order to satisfy the defined stationary distribution.

In our proposed method, some stages are inspired from our previously presented method called DBN-MBOA [19] in order to improve the method. DBN-MBOA [19] is a memory-based BOA that uses prior knowledge certainly and, therefore, like other memory-based evolutionary algorithms, suffers from the misguidance problem. Here, we attempt to resolve that problem by using the prior knowledge probabilistically.

The rest of the paper is organized as follows. Section 2 reviews the previous studies conducted on memory-based dynamic optimization. Section 3 briefly reviews the BOA and the DBN-MBOA. In Section 4, the stochastic process of “transition of BOA between consecutive generations” is modeled as “the movements among the states of a heterogeneous Markov chain”; thus, we control the movement of BOA among the states of Markov chain to address optimization in dynamic environments. Section 5 describes in detail the new method to bias the transition of BOA in the Markov chain along with its mathematical relations. The section also discusses the characteristics of the Markov chain in the proposed algorithm. The experimental results are demonstrated in Section 6; finally, Section 7 concludes the paper with a summary.

2. Literature review

This section reviews the previous studies on memory-based dynamic optimization. The first work, univariate marginal distribution algorithm [29], the simplest version of EDAs, was applied to dynamic optimization problems [49]. In [49], the best samples created by the probability vector of univariate marginal distribution algorithm are stored in the memory, which is re-evaluated in every generation. If the fitness of any memory element varies, the environment is detected to be changed. The memory is then merged with the current population to form a new population. Another approach that employed univariate marginal distribution algorithm was presented in [51], where the probability vectors of the algorithm are stored in the memory together with the best solutions. Each memory element consists of two parts: the best solution and the probability vector. The best solutions stored in the memory are re-evaluated in every generation. If environmental change is detected, the probability vector of the best memory element (i.e., the memory element with the highest fitness) is retrieved. Then, new candidate solutions to the problem are created from the probability vector, and these solutions are merged with the current population. In [53], another EDA, called population-based incremental learning [1], was used in memory-based dynamic optimization. Similar to the univariate marginal distribution algorithm, population-based incremental learning uses a probability vector. The learning process of this algorithm is conducted using an incremental process; in every generation, the probability vector is updated toward the best individual in the population. As in [51], the best solutions and the probability vectors are stored together in the