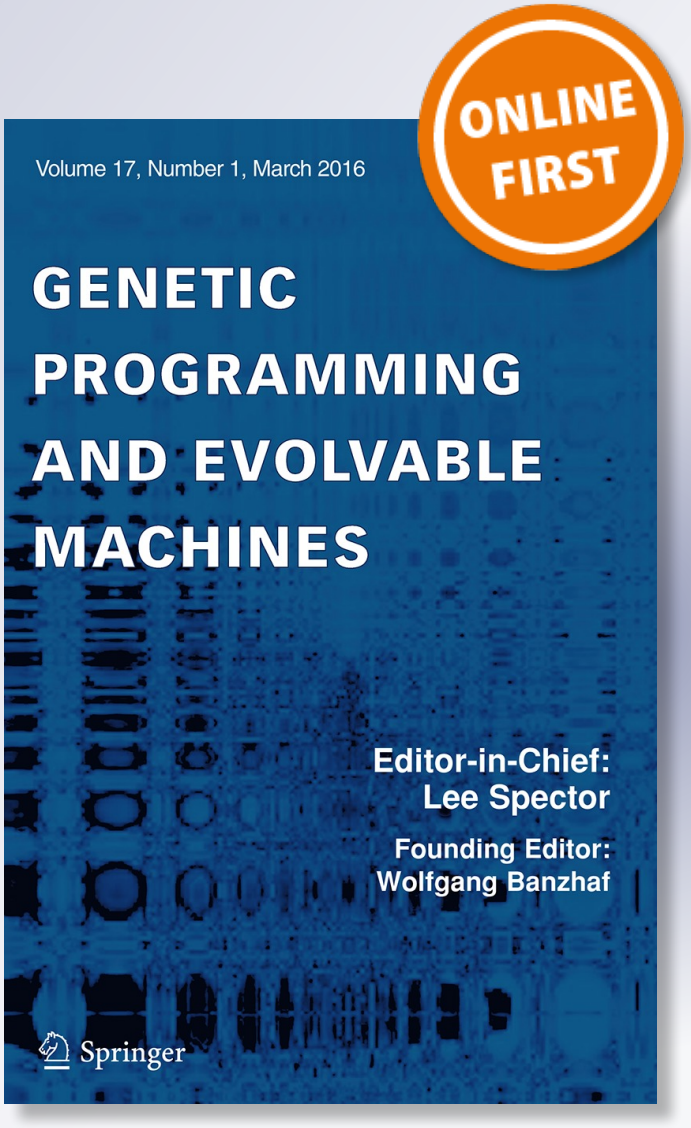
A new real-coded stochastic Bayesian optimization algorithm for continuous global optimization

Behnaz Moradabadi, Mohammad Mahdi Ebadzadeh & Mohammad Reza Meybodi

Genetic Programming and Evolvable Machines

ISSN 1389-2576

Genet Program Evolvable Mach
DOI 10.1007/s10710-015-9255-3



Springer

Genet Program Evolvable Mach
DOI 10.1007/s10710-015-9255-3

A new real-coded stochastic Bayesian optimization algorithm for continuous global optimization

Behnaz Moradabadi[1] · Mohammad Mahdi Ebadzadeh[1] · Mohammad Reza Meybodi[1]

Received: 27 September 2014 / Revised: 7 August 2015

Abstract Estimation of distribution algorithms are considered to be a new class of evolutionary algorithms which are applied as an alternative to genetic algorithms. Such algorithms sample the new generation from a probabilistic model of promising solutions. The search space of the optimization problem is improved by such probabilistic models. In the Bayesian optimization algorithm (BOA), the set of promising solutions forms a Bayesian network and the new solutions are sampled from the built Bayesian network. This paper proposes a novel real-coded stochastic BOA for continuous global optimization by utilizing a stochastic Bayesian network. In the proposed algorithm, the new Bayesian network takes advantage of using a stochastic structure (that there is a probability distribution function for each edge in the network) and the new generation is sampled from the stochastic structure. In order to generate a new solution, some new structure, and therefore a new Bayesian network is sampled from the current stochastic structure and the new solution will be produced from the sampled Bayesian network. Due to the stochastic structure used in the sampling phase, each sample can be generated based on a different structure. Therefore the different dependency structures can be preserved. Before the new generation is generated, the stochastic network's probability distributions are updated according to the fitness evaluation of the current generation. The proposed method is able to take advantage of using different dependency structures through the sampling phase just by using one stochastic structure. The experimental

✉ Behnaz Moradabadi
moradabadi@aut.ac.ir

Mohammad Mahdi Ebadzadeh
ebadzadeh@aut.ac.ir

Mohammad Reza Meybodi
mmeybodi@aut.ac.ir

[1] Department of Computer Engineering, Amirkabir University of Technology, Tehran, Iran

Published online: 04 April 2016

results reported in this paper show that the proposed algorithm increases the quality of the solutions on the general optimization benchmark problems.

Keywords Evolutionary algorithms · Estimation of distribution algorithms · Bayesian optimization algorithms · Bayesian networks

1 Introduction

Genetic algorithms are natural stochastic optimization algorithms which in each generation a set of promising solutions will be selected according to some fitness evaluations and the new generation will be produced using promising solutions, and recombination (crossover) and mutation operators [1, 2].

Estimation of distribution algorithms (EDAs) are defined as a subset of evolutionary algorithms which estimate a probability distribution model from the set of candidate solutions to help a given problem's search space [3]. Like GAs, these start with a set of promising solutions and create a probabilistic model according to the solutions that are selected. The new generation is sampled from the produced model. EDAs will do these steps over and over until a stopping criterion is satisfied. The main different between EDAs and GAs is the fact that EDAs use probabilistic models rather than the crossover and mutation operators in order to sample new solutions. Mutation and recombination operators in the GAs regularly result in the absence of using building blocks and lead to a convergence to local optima [2, 3]. Building blocks are the sub-problems of the optimization problem in terms of the interrelations among variables. Such algorithms are effectively applied to many optimization problems. EDAs have many applications in different areas [4–8]. For further reading on the application of EDAs please refer to [9].

There are two major categories of EDAs: discrete EDAs and real-valued EDAs [9]. For instance, extended compact genetic algorithm (ECGA) [10], factorized distribution algorithm (FDA) [11], and Bayesian optimization algorithm (BOA) [12–18] for discrete-value variables as well as the estimation of Gaussian networks algorithm (EGNA) [19], mixed Bayesian optimization algorithm (MBOA) [20], (mixed) iterative density-estimation evolutionary algorithms (mIDEAs) [21], real-coded Bayesian optimization algorithm (rBOA) [22] and Bayesian networks as well as Gaussian mixture model (BNGMM) [23] for real-value variables are offered.

The Bayesian optimization algorithm (BOA) is one instance of EDAs which uses Bayesian networks for modeling the promising solutions [12]. A Bayesian network is a kind of acyclic directed graph with one node for each variable in the optimization problem, while the edges indicate the conditional dependencies among variables. In the BOA, a Bayesian network is built from the set of promising solutions and the new solutions will be sampled from the built Bayesian network. The experimental results of the BOA show that it is able to identify correct linkages among the optimization problems' variables.

This paper proposes a novel stochastic Bayesian network with the aim to obtain a novel real-coded stochastic Bayesian optimization algorithm (rsBOA) for continuous global optimization. In the rsBOA, the Bayesian network is characterized by a

stochastic structure. The stochastic Bayesian network exhibits a probability distribution function for each possible edge within the network and such stochastic structure is used for sampling the new generation. After generating new generation, the fitness of the population will be evaluated and the probability distributions of the stochastic Bayesian network updated according to the fitness evaluations of the current generation. So, in the rsBOA, various structures of dependencies are preserved only inside one stochastic structure. To generate a new solution during the model sampling phase, a new structure is sampled according to each edge's probability distribution function. Following that, a new solution will be sampled from the generated Bayesian network. The proposed rsBOA is capable of taking advantage of using different structures during the sampling phase since each new solution can be sampled according to a different structure. Therefore, the proposed algorithm is able to search the problem space for a variety of dependency structures and to realize the building blocks.

The rest of the paper is organized in the following way. Section 2 will explain the related works on the continuous EDAs. Section 3 will introduce stochastic Bayesian network. Section 4 will present the proposed real-coded stochastic Bayesian optimization algorithm (rsBOA) and Sect. 5 reports the experimental results obtained with the proposed algorithm. Finally, the paper ends with a conclusion in Sect. 6.

2 Related works

Here we will review the related works about the real-coded EDAs. In EGNA, a Gaussian network is created in each generation using a scoring metric and the new generation is sampled from the network that is constructed [19]. The EGNA uses only one single-peak Gaussian model, therefore not appropriate for complex optimization problems [19]. The IDEAs make use of the joint normal kernels distribution, in which each selected solution is surrounded by a single normal distribution. Each normal distribution's variance may be either fixed to a relatively small value, but it must be preferable to adapt variances based on the current search status. The mixed Bayesian optimization algorithm (mBOA) models continuous variables vectors and it uses an extension of Bayesian networks with local structures [20]. A model used in mBOA is composed of a decision tree for each of the variables. In a decision tree, the leaves will correspond to the sub regions of the search space and each leaf models a single-variable mixture of normal distributions as centered on the correspond variable's values.

The mixed IDEAs are an extension of IDEAs which use some mixture of normal distributions as a way to model each of the variables. The mIDEAs cluster the candidate solutions to utilize a mixture of normal distributions [21]. In the sampling phase then, mIDEAs select a distribution (or cluster) of a mixture of normal distributions and thereby generate the new individuals. Thus, these algorithms are not able to realize the building block except that one cluster includes most of the building blocks.

In rBOA, the promising solutions will be selected from the current population by using a parent selection method, and a Bayesian network is then constructed in order to model the promising solutions' distribution [22]. The new individuals are then sampled from the constructed Bayesian network through the following procedure. In the sampling phase, the sub-trees (building blocks of the problem) of the Bayesian network will be extracted. Then, the entire solutions of each sub-tree are clustered to model with a mixture of normal distributions. The new partial solutions of each sub-tree are then sampled from the related mixture of normal distributions. Finally, the new individuals will be merged into the main population and replaced some of the old ones. The mentioned steps are repeated up to the time when some stopping criterion is satisfied [22]. Unlike mIDEAs, which recognize a mixture of normal distributions from the population in the model sampling phase, rBOA make use of a mixture of normal distributions of each building block at that stage. Therefore, rBOA can do better preserving the building blocks as compared to the mIDEAs in continuous optimization problems.

Bayesian networks and Gaussian mixture model (BN-GMM) is an evolutionary algorithm for continuous optimization which uses a Bayesian network for modeling the dependencies among the variables of the problem. Then, one Gaussian mixture model will be adopted to model the probability distribution for each sub-problem and the new generation will be produced from the Gaussian mixture model of each sub-problem [23].

In [24–26], we have proposed some improved version of real-coded Bayesian optimization algorithm, called IrBOA. The IrBOA utilizes an adaptive clustering method to break the problem space into multiple Bayesian networks. Each Bayesian network has its own structure and parameters. The adaptive clustering method leads to searching the problem space efficiently in the initial generations of the algorithm. Also, it improves the quality of the solutions by realizing the building blocks in the following generations of the algorithm. The utilization of adaptive clustering brings about the realization of the building blocks; also when the diversity of solutions is decreased, the number of clusters will be decreased in the same manner. Therefore, a smaller number of Bayesian networks are generated and more building blocks will be amassed in one Bayesian network [25, 26].

In our previous work [27], we have offered a new real-coded Bayesian optimization algorithm which is based on a team of learning Automata (LA-rBOA) that uses a team of learning automata in order to construct the Bayesian network. There, there is one learning automaton for each possible edge in the Bayesian network, and each learning automaton tries to learn the existence or non-existence of the corresponding edge. In each generation, one Bayesian network will be constructed according to the actions of the learning automata. The next generation is then sampled from the built Bayesian network and each learning automaton then updates its action probability based on the quality of the Bayesian network that has been built. The experimental results indicate that the algorithm proposed is superior to other related algorithms in terms of quality and performance measures.

Chen and Chen [28] proposed a quality measure of discretization method for EDAs and utilizes it to analyze fixed-width histogram (FWH), fixed-height histogram (FHH), and greedy random split (GRS). Then they integrated rBOA with

FWG, FHH, and GRS. The experimental results show that using these discretization methods in the rBOA improves the continuous global optimization using standard benchmark test suite.

In many EDAs, generating the dependency model of the problem variables is seen as a bottleneck; for example finding the best structure of a Bayesian network is an NP-hard problem [29]. Also, in many optimization problems, the dependency structures of each sub-space can differ from other sub-spaces. Thus, the chance to use some techniques for generating different structures with an acceptable complexity is a valuable context in EDAs.

3 Stochastic Bayesian networks

As mentioned before, the BOA constructs a Bayesian network by using a set of promising solutions and samples the next generation from the constructed Bayesian network [22]. In the following of this section we introduce a new structure for Bayesian network that is called stochastic Bayesian network: A stochastic Bayesian network $\mathbf{M} = (\boldsymbol{\zeta}, \boldsymbol{\theta})$ consists of a stochastic structure $\boldsymbol{\zeta}$ and a set of parameters $\boldsymbol{\theta}$. The structure of the network is described by a stochastic graph where each edge has a probability of existence. Because of stochastic structure, we must sample a new graph from the stochastic structure to construct a Bayesian network. This stochastic structure is designed such that if the probability of an edge will be 1, this edge appears in the sampled graph. On the other hand if the probability of an edge will be 0 then the corresponding edge should not be appeared in the sampled graph. Finally in the sampled graph there is a set of nodes that are corresponding to the problem variables and a set of edges that are corresponding to the conditional dependencies between variables. The parameter $\boldsymbol{\theta}$ is a set of probability distributions that is described as a product of probability density functions for the edges that are appeared in the network. Generally, the procedure of finding a probabilistic model has two steps: first, a procedure to construct the structure and second, a scoring metric that evaluates the built structure [28–32]. In this paper we propose a new rBOA that uses a stochastic Bayesian network to generate the probabilistic model of promising solutions. In the next section we will describe the scoring metric and the search procedure that are used to learn the proposed stochastic Bayesian network.

4 The proposed real-coded stochastic bayesian optimization algorithm

This section of the paper proposes a new real-coded stochastic Bayesian optimization algorithm (rsBOA) for continuous global optimization. The design of this algorithm is based on a stochastic Bayesian network. This stochastic Bayesian network has a probability distribution function for each possible edge in the network. The algorithm proposed here starts with some random population. The probability of each edge at the start-up of the algorithm is set to l/d, $p_i = 1/\mathrm{d}$ where l is the assumed maximum number of possible links for each variable, and d is the number of variables. In the model building phase, in each generation of the rsBOA a

set of promising solutions is selected with regard to fitness evaluations, and after that a tree Bayesian network (a Bayesian network with tree structure) that maximizes the data likelihood is calculated for the set of promising solutions. In order to find a tree Bayesian network, the rsBOA makes use of the Chow-Liu algorithm that generate a tree that maximizes the data likelihood [33]. After that, the probability distribution function of each edge in the stochastic structure will be updated in accordance with the existence of the corresponding edge in the tree Bayesian network by utilizing an updating rule. In the model sampling phase of rsBOA, to sample a new solution, a new Bayesian network will be sampled from the current probabilistic model. This Bayesian network exhibits its own structure and parameters. Thus, each new solution within the new population can be sampled from a different structure. After generating the new population, the fitness evaluation will be applied on the new population and the new population will merge into the old one. Finally, the new generation is selected through a replacement

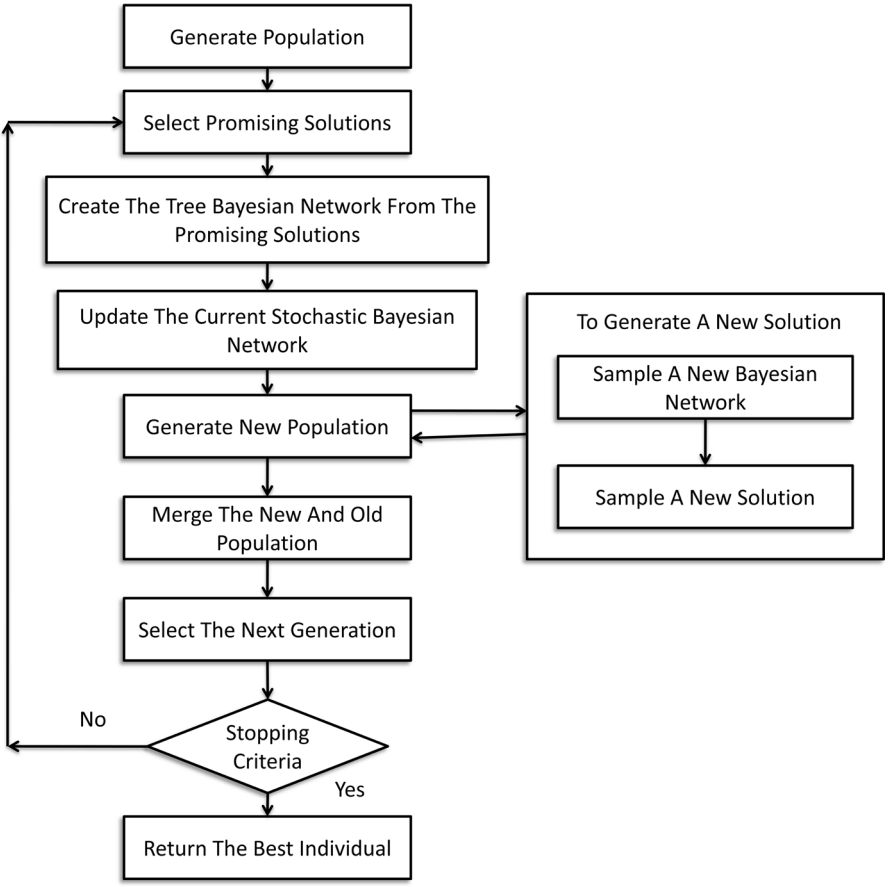


Fig. 1 The pseudo code of the proposed algorithm (rsBOA)

method. These procedures are repeated until some stopping criterion is satisfied. The overall process of the rsBOA can be seen in Fig. 1. The proposed rsBOA can take advantage of different structures in the sampling phase such that, each new solution can be sampled from a unique structure. So, the proposed algorithm is capable of searching the problem space for a variety of dependency structures as well as realizing the building blocks only by using one stochastic structure. You can see the pseudo-code of the rsBOA in Algorithm 1. In the following sections the model building and the model sampling phases will be described in detail.

Algorithm 1: Pseudo code of the proposed algorithm (rsBOA)

Let d be the dimension of the optimization problem, Np be the population size and l be the assumed maximum number of possible links for each variable;
Let t be the generation counter and initially set to 0 and $\mathrm{Max}_{\mathrm{NFC}}$ be the total number of generations.
Create a random population P of Np individual.
Set the initial probability distribution function of each possible edge j to be $p_j = l/d$.
While $t < \mathrm{Max}_{\mathrm{NFC}}$ do
    Select τ % individuals S from population P based on the fitness evaluations.
    Generate the tree Bayesian network using Chow-Liu algorithm.
    Update the probability distribution function of each p_j based on the network that is generated in the previous step.
    Sample m new Bayesian networks using the probability distribution functions.
    Remove the cycles in the Bayesian networks using DFS algorithm.
    For n=1 **to** m **do**
        Extract all trees (sub-problems) in the Bayesian network indexed by r.
        For j=1 **to** r **do**
            Generate the $\frac{Np}{m}$ partial solutions for $\frac{Np}{m}$ individuals based on the conditional normal distribution of promising solutions
        End for
            Merge all partial solutions together to create a new individual.
    End for
    Merge the new generation and the old generation.
    Evaluate population based on the fitness function and choose the best Np individuals as the new generation.
    Set t=t + 1
End while

4.1 Model building

Throughout the following sub-sections we will describe the scoring metric and the search procedure that are used in the proposed algorithm.

4.1.1 Scoring metric

There are a variety of metrics proposed for the evaluation of Bayesian networks. For instance, Bayesian metrics measure the quality of the Bayesian network through computing a marginal likelihood of the Bayesian network with respect to the given data and the inherent uncertainties [29–31]. Minimum description length metrics operate on the assumption that the number of regularities in the data that are

encoded by a model is somewhat proportional to the amount of data compression that the model allows [32]. The Bayesian information criterion (BIC) is some criterion for model selection within a limited set of models. It works based on the likelihood function [30]. This paper uses the Chow-Liu algorithm which tries to estimate the optimal dependence tree for the given problem. In the Chow Liu algorithm, there is a complete undirected graph G of d nodes where d is number of variables in the optimization problem that is concerned [33]. If x_i and x_j are taken to be the variables of the optimization problem, the weight of edge(i, j) is then calculated following the equation below:

$$edge(i,j) = \sum_{x_i,x_{j(i,j)\in D}} P(i,j)\log\frac{P(i,j)}{P(i)P(j)} \tag{1}$$

here $P(i, j)$ is the joint probability distribution of x_i and x_j. Also, $P(i)$ and $P(j)$ are the probability distributions of x_i and x_j, respectively. After calculating the weight of each edge(i, j), the maximum spanning tree of graph G is calculated as the problem's dependence tree [19].

4.1.2 Search procedure

As was mentioned before, the problem of learning a Bayesian network's structure according to some scoring metric is an NP-complete problem. Therefore, there is no known polynomial-time algorithm for finding the best network structure that corresponds to most scoring metrics [29]. To overcome this problem two simple methods are often used:

1. A simple, greedy algorithm is used in order to build the network structure [30, 31]: The greedy algorithm will add an edge with the greatest improvement in the current network quality in each step until no more improvement is possible. Since Bayesian networks are acyclic graphs, the graph structure has to be validated following each operation [22, 34, 35]. The initial network structure can be some graph with no edges. It can also take advantage of the prior information such as the best tree computed by the polynomial-time maximum branching algorithm [9, 30].
2. The structure of the Bayesian network is limited to a simpler one (such as tree or chain) that can be constructed optimally or near optimally in a polynomial time. For more information about these limited models please refer to [9, 12].

In the algorithm proposed here, the structure of the Bayesian network is defined as the following: Let d be the dimension of the real-valued optimization problem. Thus, the corresponding Bayesian network has d nodes. Because the Bayesian network is a directed graph, the maximum number of edges in the network with d nodes will be $D = d \times (d - 1)$. For each edge j at time t, there will be a probability distribution $p_j(t)$ which determines the existence probability of the corresponding edge in the Bayesian network. Probability $p_j(t) = 0$ means that the corresponding edge does not appear in the Bayesian network, while value $p_j(t) = 1$ indicates that

the corresponding edge occurs in the network. Since there is no prior information about the dependencies of problem variables at the first iteration, we will use the uniform probability distribution for each edge, $p_j(0) = 1/\mathrm{d}\ (for\, j = 1, 2, \ldots, d)$. The initial probability vector of each edge j can also be configured according to the prior information on the variables dependency of the problem.

The main goal by using the stochastic structure in the rsBOA is that the stochastic structure is able to model multiple structures in one model. So, different structures are able to be sampled from the main structure. This characteristic enables us to model different dependency structures at a time and further to sample new solutions through the different structures. The main problem about such a structure is how the probability distribution function of each edge is updated in the structure. In order to solve this challenge we will use the following procedure: in each iteration of the rsBOA, the $\tau\%$ of the population is selected according to the truncation selection as the candidate solutions. Then a dependence tree is built for this set of promising solutions by using the Chow-Liu algorithm which was described earlier in this section. In order to calculate the probabilities of Eq. (1) in the form of a continuous domain, we model the optimization problem's probability distribution by a Gaussian mixture of normal distributions. When the dependence tree of the candidate solutions is found, the probability distribution function for each edge j will be updated according to the existence or lack of edge j in the generated dependence tree according to the following equation:

$$\begin{cases} p_j(t+1) = Max\big(p_j(t) + \lambda p_j(t), 1\big) & edge(j) \in Dependence\,Tree \\ p_j(t+1) = Min\big(p_j(t) - \lambda p_j(t), 0\big) & edge(j) \notin Dependence\,Tree \end{cases} \tag{2}$$

in which $p_j(t)$ is denoted as the probability distribution function for edge j at time t and λ is a constant factor. $p_j(t+1)$ is updated in a way that it remains a probability distribution function. Therefore, if edge j exists in the dependence tree, the probability distribution function of edge j, $\mathrm{p_j}$, must be increased; by contrast, in the case that edge j does not exist in the generated dependence tree, the probability distribution function of edge j, $\mathrm{p_j}$ must be decreased. After updating the probability distribution function for each edge in the network, we can sample a new Bayesian network structure by using this information. To determine whether an edge occurs in the Bayesian network, the search procedure will use the probability distribution function of the corresponding edge. By this time the constructed Bayesian network structure may have some cycles. In order to remove the cycles, the search procedure will use a depth first search algorithm (DFS) with a random start node in order to navigate the current structure and mark the entire back edges which point from one node to one of its ancestors. After it has detected the back edges, the search procedure will remove all cycles. Finally, this network is used for sampling a new solution at the sampling phase.

4.2 Model sampling

The proposed algorithm should generate Np new solutions in the model sampling phase. With the aim to generate N new solutions through different structures with some acceptable complexity, we will sample m Bayesian networks from the

stochastic model that is described in the previous sections. Following that, for each Bayesian network $\frac{Np}{m}$ solutions must be generated. This sub-set of solutions are then merged together to produce a set of *Np* solutions. In order to generate a new solution from a Bayesian network, all maximal connected sub-graphs are first extracted from the constructed Bayesian network. These sub-graphs will determine the sub-problems whose variables has interactions. With the aim to simplify the presentation, the promising solutions related to a sub-problem are called partial solutions of that sub-problem. In order to generate a partial solution that corresponds to sub-problem *r*, the rsBOA will employ a mixture of Gaussian normal distributions to the selected partial solutions of the set of promising solutions and it then generates the new partial solutions from the distribution [22, 26, 27]:

$$f(Y_0|Y_1, Y_2, \ldots, Y_n) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(Y_0-\mu_{i0})^2}{2\sigma_{i0}^2}} \tag{3}$$

where Y_0 is the partial solution which must be produced, Y_1, Y_2, …, Y_n are parents to Y_0, and i represents the selected normal distribution of the Gaussian mixture model; and

$$\sigma_{i0} = \frac{1}{\sqrt{\left(\Sigma^{-1}\right)_{0,0}}}, \tag{4}$$

$$\mu_i = \mu_{i0} - \frac{\Sigma_{j=1}^{Y_n}\left(Y_j - \mu_j\right)\left(\Sigma^{-1}\right)_{j,0}}{\left(\Sigma^{-1}\right)_{0,0}}, \tag{5}$$

where μ_{i0} is the mean of $\mathrm{Y_j}$ and Σ^{-1} represents the inverse of covariance matrix for $(\mathrm{Y_0}|\mathrm{Y_1}, \mathrm{Y_2}, \ldots, \mathrm{Y_n})$ of the selected solutions.

5 Experimental results

In this section, the performance of the proposed algorithm has been evaluated through some computer experiments conducted. The performance of the proposed algorithm will be compared with some related algorithms on the standard benchmark test functions. In each of the experiments we consider the quality of solutions, computation time, and convergence rate as criteria for performance. In the initial experiment the performance of the proposed algorithm will be compared with some related rBOA algorithms. In the second one the performance of the proposed algorithm will be compared with some related evolutionary algorithms.

5.1 Experiment 1: Comparison of the rsBOA with some related rBOA algorithms

In this section, we will compare the outcome of the proposed rsBOA with some related rBOA. In the first subsection, the used benchmark functions will be

Table 1 The CEC2005 benchmark functions

Function	Function name	Range	Optimal value	Property
F1	Shifted Sphere Function	[−100 100]	−450	Uni-modal Shifted Separable
F2	Shifted Schwefel's Problem	[−100 100]	−450	Uni-modal Shifted Non Separable
F3	Shifted Rotated High Conditioned Elliptic Function	[−100 100]	−450	Uni-modal Shifted Non Separable Rotated
F4	Shifted Schwefel's Problem 1.2 with Noise in Fitness	[−100 100]	−450	Uni-modal Shifted Non Separable
F5	Schwefel's Problem 2.6 with Global Optimum on Bounds	[−100 100]	−310	Uni-modal Non Separable
F6	Shifted Rosenbrock's Function	[−100 100]	390	Multi-modal Shifted Non Separable
F7	Shifted Rotated Griewanks Function without Bounds	[0 600]	−180	Multi-modal Shifted Non Separable Rotated
F8	Shifted Rotated Ackleys Function with Global Optimum on Bounds	[−32 32]	−140	Multi-modal Shifted Non Separable Rotated
F9	Shifted Rastrigins Function	[−5 5]	−330	Multi-modal Shifted Separable Rotated
F10	Shifted Rotated Rastrigins Function	[−5 5]	−330	Multi-modal Shifted Non Separable Rotated
F11	Shifted Rotated Weierstrass Function	[−0.5 0.5]	90	Multi-modal Shifted Non Separable Rotated
F12	Schwefel's Problem 2.13	[$-\pi$ π]	−460	Multi-modal Shifted Non Separable

Table 1 continued

Function	Function name	Range	Optimal value	Property
F13	Expanded Extended Griewank's plus Rosenbrock's Function	[−3 1]	−130	Multi-modal Shifted Non Separable
F14	Shifted Rotated Expanded Scaffer's F6	[−100 100]	−300	Multi-modal Shifted Non Separable

introduced and in the second subsection, the results of comparison between the proposed rsBOA and other related algorithms will be presented. In the third and fourth experiments the convergence rate and computation time of the proposed algorithm are compared with those of other related algorithms. In the fifth experiment then the parameter sensitivity of the proposed algorithm will be analyzed.

5.1.1 CEC2005 test functions

This set of test functions is chosen from the benchmark functions that are used in CEC2005 [36] as presented in Table 1. This set includes five uni-modal (F1–F5) plus nine multi-modal (F6–F14) functions and the quality of the solution is used as the measure for performance. Each function in this test suite stands for a real-value function which must be minimized. Information related to this test suite is presented in Table 1. For additional information on this benchmark please see the Ref. [36]. The experiment in this section is conducted based on the assumption that dimension $d = 10$ and the parameter m (The number of Bayesian networks that is sampled from the stochastic model) is set to 10.

5.1.2 The comparison of optimization results

In this part, rsBOA is compared with the original rBOA [22], the learning automata based rBOA (LA-rBOA) [27] and the improved real-coded rBOA (IrBOA) [26] and BOA integrated with GRS discretization (GRS-BOA) [28] on the test functions which was introduced in earlier section. Based on No Free Lunch theorem: "for any algorithm any elevated performance over one class of problems is offset by performance over another class" [37]. Thus, the rsBOA tries to solve a class of benchmark functions in terms of accuracy and efficiency. For the entire conducted experiments the parameters of the rsBOA are chosen empirically according to the sensitivity analysis which is given in the last sub-section. Also, two points have to be reminded: (1) the initial population in all algorithms is created according to the uniform distribution on the boundary values for each function, and (2) in order to make a comparison, the maximum number of function calls, Max_{NFC}, will be set to 100,000.

Table 2 Numerical results for rsBOA, Original rBOA, LA-rBOA and IrBOA

Function	Original rBOA		IrBOA		LA-rBOA		GRS-BOA		rsBOA	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
F1	−4.40E+02	2.55E−09	−4.42E+02	6.47E−11	−4.45E+02	6.89E−11	**−4.50E+02**	0.00E+00	**−4.50E+02**	1.23E−15
F2	−4.14E+02	3.61E−08	−4.17E+02	4.16E−10	−4.16E+02	1.14E−11	−4.45E+02	1.74E+01	**−4.50E+02**	1.51E−16
F3	−4.17E+02	4.72E−36	−4.19E+02	7.25E−12	−4.21E+02	2.04E−36	1.06E+06	1.06E+12	**−4.50E+02**	6.41E−36
F4	−4.16E+02	7.31E−25	−4.18E+02	3.20E−12	−4.20E+02	2.76E−27	−4.07E+02	3.42E+03	**−4.50E+02**	4.72E−29
F5	−3.14E+02	3.16E−14	−3.02E+02	9.03E−12	−3.03E+02	6.76E−17	−1.94E+02	5.98E+04	**−3.07E+02**	7.42E−19
F6	3.92E+02	7.26E−22	4.02E+02	8.55E−12	3.95E+02	6.29E−18	4.91E+02	5.13E+04	**3.89E+02**	2.71E−36
F7	−1.57E+02	4.27E−13	−1.62E+02	7.75E−14	−1.75E+02	4.70E−15	−1.08E+03	1.38E−09	**−1.82E+02**	4.99E−27
F8	**−1.39E+02**	2.85E−10	**−1.39E+02**	1.37E−12	−1.37E+02	1.81E−11	−1.19E+02	1.37E−03	**−1.39E+02**	2.68E−22
F9	−3.06E+02	6.72E−14	−3.04E+02	2.64E−09	−3.04E+02	2.63E−12	−3.28E+02	1.70E+00	**−3.30E+02**	6.21E−29
F10	−3.05E+02	7.25E−13	−3.02E+02	7.03E−07	−3.03E+02	2.51E−12	**−3.23E+02**	4.46E+00	−3.09E+02	5.67E−29
F11	1.03E+02	5.19E−12	1.00E+02	9.80E−02	9.80E+01	1.20E−15	**9.47E+01**	3.22E+00	9.80E+01	9.32E−23
F12	−4.50E+02	1.73E−10	−4.50E+02	3.52E−08	−4.52E+02	4.67E−15	1.18E+02	4.07E+05	**−4.56E+02**	3.65E−19
F13	−1.20E+02	4.99E−12	−1.22E+02	5.73E−12	−1.23E+02	5.83E−16	−1.29E+02	8.58E−02	**−1.30E+02**	3.89E−18
F14	−2.83E+02	1.18E−10	−2.85E+02	9.88E−14	−2.88E+02	3.57E−15	−2.96E+02	1.37E−01	**−2.99E+02**	4.15E−17

In this experiment, the parameter settings for Original rBOA, LA-rBOA and IrBOA are borrowed from [26, 27]. Also the result of GRS-BOA are borrowed from [28]. The results for the rsBOA, Original rBOA, LA-rBOA, GRS-BOA, and IrBOA on the used test functions are summarized in Table 2. For each function the mean error values as well as the standard deviations of the obtained fitness values on the 50 independent runs are presented. It must be mentioned that for the tables given down in this section, the best results are highlighted. Also, to have a better comparison, a set of two-tailed t tests were given; because the t test clarifies the difference between the means of two independent samples. The t value to check if the two algorithm means are different can be calculated by the following equation:

$$t\,\mathrm{test}(\mathrm{A1},\mathrm{A2}) = \frac{\hat{\mathrm{X}}_1 - \hat{\mathrm{X}}_2}{\sqrt{\frac{\mathrm{s}_1^2}{\mathrm{n}_1} + \frac{\mathrm{s}_2^2}{\mathrm{n}_2}}}, \tag{6}$$

in which $\hat{\mathrm{X}}_1$, s_i^2 and n_i represent the means, standard deviation, and the number of runs for algorithm i, respectively. In this experiment we consider the rsBOA as algorithm $A2$. The result of t value and p value (with $\alpha = 0.05$ and degress of freedom $= 98$) among the experiments obtained with the rsBOA, Original rBOA, LA-rBOA, GRS-BOA, and IrBOA for the test functions are shown in the Table 3. Tables 2 and 3 indicate that in the most used test functions, rsBOA clearly outperforms Original rBOA, LA-rBOA and IrBOA. Also the result show that rs-rBOA outperforms GRS-BOA in most of used test function.

In what following the analysis between the rsBOA and the two older rBOA (IrBOA and LA-rBOA) is described: Function F1 is shifted Sphere function, function F2 is shifted Schwefel's problem 1.2, and function F3 is a shifted rotated high conditioned elliptic one. These three functions have different conditions which make F3 to be harder than F2 and F2 to be harder than F1. Based on the results we can conclude that rsBOA outperforms the three other algorithms for functions F1, F2 and F3. Function F4 is shifted Schwefel's problem 1.2 with noise in fitness. Noise in the fitness disturbs the search process. From the results, we observe that the proposed rsBOA is superior to the Original rBOA, IrBOA and LA-rBOA. Function F5 is Schwefels problem 2.6 with global optimum on bounds. For this function, the proposed algorithm is superior to the Original rBOA, IrBOA and LA-rBOA. Function F7 is a shifted rotated Griewanks function without bounds, just the initialization range is given. Griewanks function is more difficult with low dimension and it is also difficult to achieve the global optimum. The rsBOA gains better result in comparison to the Original rBOA, IrBOA and LA-rBOA. Function F8 is shifted rotated Ackleys function with global optimum on bounds which has a very narrow global basin half of the dimensions of which are on the bounds. Thus, the search looks almost like seeking a needle in a haystack. For this function, the results of the rsBOA are clearly better than IrBOA but the result of rsBOA is a little better than Original rBOA and LA-rBOA. Functions F9 and F10 are shifted Rastrigins function and shifted rotated Rastrigins function respectively. The two have a huge number of local optima. The results indicate equal performance for rsBOA in comparison to the Original rBOA, IrBOA and LA-rBOA on function F9

Table 3 Outcomes of statistical test for the rsBOA, Original rBOA, LA-rBOA, the IrBOA, and GRS-BOA

Function	Original rBOA		IrBOA		LA-rBOA		GRS-BOA	
	t value	*p* value	*t* value	*p* value	*t* value	*p* value	*t* value	*p* value
F1	2.77E+10	<0.0001	8.74E+12	<0.0001	5.13E+11	<0.0001	0.00E+00	=1
F2	7.05E+09	<0.0001	5.60E+12	<0.0001	2.11E+13	<0.0001	2.03E+00	<0.05
F3	2.93E+37	<0.0001	2.73E+37	<0.0001	3.05E+37	<0.0001	7.07E−06	=1
F4	3.29E+26	<0.0001	7.05E+26	<0.0001	7.68E+28	<0.0001	8.89E−02	<1
F5	−1.57E+15	<0.0001	3.91E+16	<0.0001	4.18E+17	<0.0001	1.34E−02	<1
F6	2.92E+22	<0.0001	1.07E+15	<0.0001	6.75E+18	<0.0001	1.41E−02	<1
F7	4.14E+14	<0.0001	2.48E+16	<0.0001	1.05E+16	<0.0001	4.63E+07	<0.0001
F8	0.00E+00	=1	0.00E+00	=1	7.81E+11	<0.0001	1.03E+05	<0.0001
F9	2.53E+15	<0.0001	5.28E+13	<0.0001	6.99E+13	<0.0001	8.32E+00	<0.0001
F10	3.90E+13	<0.0001	6.13E+12	<0.0001	1.69E+13	<0.0001	−2.22E+01	<0.0001
F11	6.81E+12	<0.0001	1.06E+16	<0.0001	0.00E+00	=1	−7.25E+00	<0.0001
F12	2.45E+11	<0.0001	2.91E+12	<0.0001	6.06E+15	<0.0001	9.97E−03	<1
F13	1.42E+13	<0.0001	1.35E+16	<0.0001	8.49E+16	<0.0001	8.24E+01	<0.0001
F14	9.59E+11	<0.0001	4.07E+13	<0.0001	2.18E+16	<0.0001	1.55E+02	<0.0001

and better performance on function F10. Function F11 is a shifted rotated Weierstrass function and Function F12 is Schwefels problem. For these function the rsBOA gets better solution compared to other algorithms. Finally for function F13 through F15 the rsBOA gains better result in comparison to Original rBOA, IrBOA and LA-rBOA.

The major different between IrBOA and the proposed algorithm is that IrBOA clusters the population using an adaptive clustering method and generates a Bayesian network for each cluster; but based on the nature of evolutionary algorithms, the diversity of population is decreased over time, so the number of Bayesian networks is also decreased over time. While in the proposed algorithm, we use only a stochastic Bayesian network and there is a possibility of generating different Bayesian networks until all edges probability converge to some value. In other words the proposed algorithm can preserve multi Bayesian networks in one structure a longer time than IrBOA without using any clustering method that results in a lower computational complexity and running time. On the other hand the major different between LA-rBOA and the proposed algorithm is that LA-rBOA uses a learning automaton for each possible edge in the network without any idea to preserve multi Bayesian networks in one generation. The LA-rBOA only uses a fast method to learn the Bayesian network. So, the proposed algorithm is superior to the LA-rBOA in most of used benchmark functions. Generally, according to the obtained results it is reasonable to say that the rsBOA presents evidences of better performance than similar algorithms such as Original rBOA, LA-rBOA and IrBOA.

5.1.3 The comparison of convergence speed for rsBOA

The comparison of convergence speed for eight test functions for 5 typical runs (1000 iteration) among the rsBOA, Original rBOA, LA-rBOA and IrBOA is presented in Fig. 2. Each sub-figure shows the obtained fitness value from four algorithms throughout the optimization process. It can be seen from the Fig. 2 that rsBOA has a lower convergence rate than the three other algorithms in all of used test suits.

5.1.4 The comparison of computation time for rsBOA

The rsBOA is also compared in terms of computation time with the Original rBOA, LA-rBOA and IrBOA. These algorithms are implemented in MATLAB R2009a in a PC, with a single CPU of Intel(R) Core(TM)2 Duo 3.33 GHz and a 4 GB memory. Table 4 shows the computation time for four algorithms for the used test functions. It can be seen that the rsBOA requires a smaller computation time than the three other algorithms.

5.1.5 Sensitive analysis

In this section we study the effects of each parameter on the performance of the rsBOA. The proposed algorithm has six parameters: (1) Selection strategy method, (2) Parameter τ in the selection strategy, (3) Population size (Np), (4) The number

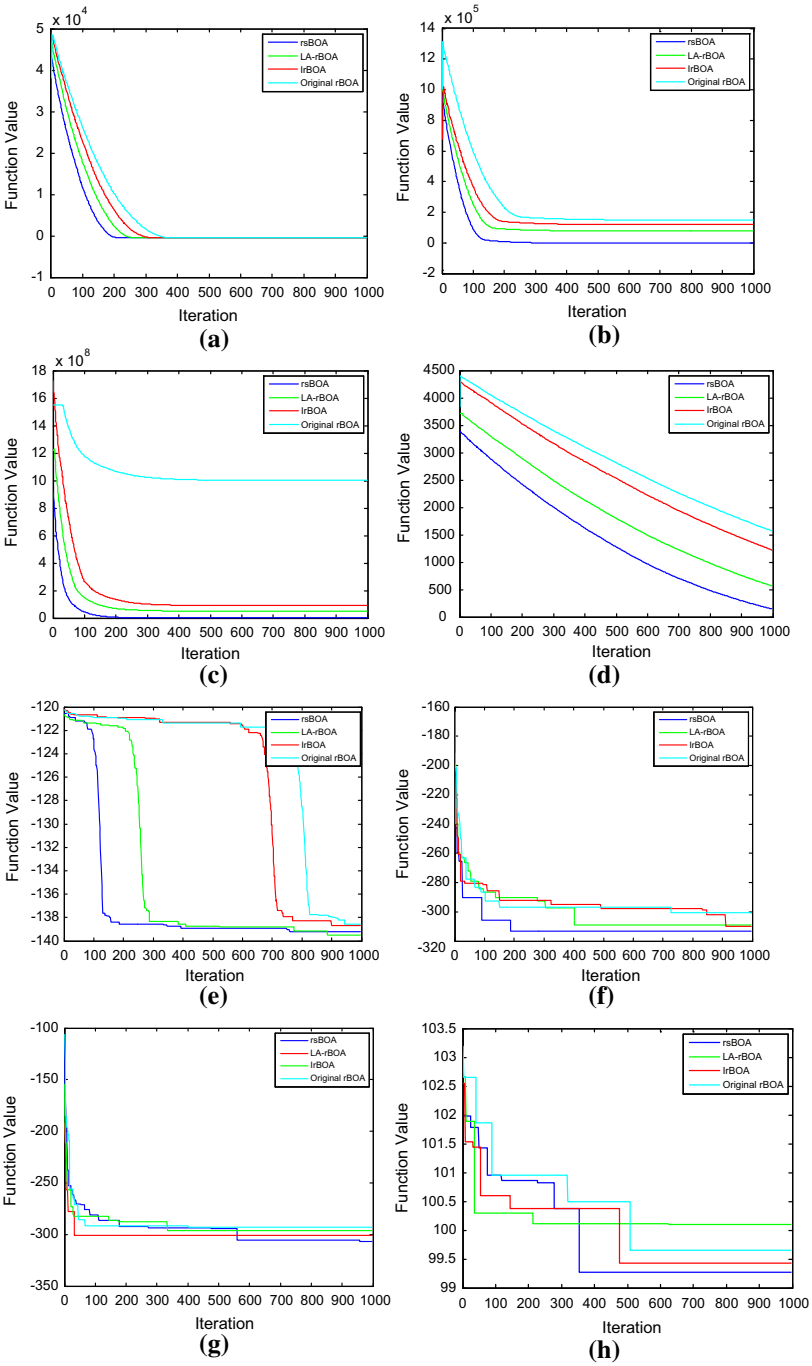


Fig. 2 The convergence speed for the rsBOA, LA-rBOA and the IrBOA. **a** Function $F1$, **b** function $F2$, **c** function $F3$, **d** function $F7$, **e** function $F8$, **f** function $F9$, **g** function $F10$, **h** function $F11$

of Mixture components (K), (5) the constant Factor in the updating rule (λ) and (6) the maximum number of possible links of each variable of the optimization problem (l). In order to study the impacts of one parameter, the values of other parameters are fixed. For the first parameter there are two common strategies: tournament selection versus truncation selection. In tournament selection, τ individuals are randomly picked from the population and the best one is then chosen as a candidate solution [1, 2]. In truncation selection the best $\tau\%$ individuals of the population are selected as candidate solutions [1, 2]. Since the updating rule of stochastic structure depends on the promising solutions' dependence tree, it is evident that in the truncation selection a larger number of better solutions are selected and it will give better results in comparison to the tournament selection. Therefore, the experiments of the proposed algorithm are performed based on the truncation selection. The default values of the parameters are listed in the following setting: (1) the parent selection method is truncation selection (2) parameter $\tau = 20$, (3) population size (Np) $= 500$, (4) the number of Mixture components in the model building and model sampling phase (K) $= 10$, (5) the constant factor in updating rule $\lambda = 0.4$, and (6) the number of links for each variable $l = 5$.

The impact of parameter τ in the truncation selection This experiment will compare the effect of parameter τ in truncation selection on the performance of the rsBOA. In order to do that, parameter τ is varied from the set $\{10\,\%, 30\,\%, 50\,\%, 70\,\%\}$ and the performance of the proposed algorithm in 100 iterations is calculated following for some of the test functions of Table 1. Table 5 exhibits the results of this experiment. Table 5 indicates that the best value of parameter τ is in range $[10\,\%30\,\%]$.

The effect of the population size (Np) This experiment compares the impact of parameter Np on the performance of the rsBOA. To do that, the value of Np is chosen from the set $\{100, 300, 500, 700, 900\}$ and the results of the proposed algorithm in 100 iterations for some of the test functions of Table 1 are given in Table 6. Table 6 shows the best value of the parameter Np in term of the improvement in the quality solutions is about 500.

The impact of the number of mixture components (K) This experiment shows the influence of the number of mixture components on the performance of the rsBOA. To do that, the value of K is chosen from the set $\{5, 10, 20, 30\}$ and the result of the proposed algorithm in 100 iterations for some of the test functions of Table 1 is represented in Table 7. This Table shows that the best value of parameter K is about 10.

The impact of constant factor in the updating rule (λ) In this experiment, the impact of parameter λ on the performance of the rsBOA on some of the used test functions is exhibited in Table 8 where it is shown that the best value of parameter λ is between 0.2 and 0.4.

The maximum number of possible links for each variable (l) Finally, the impact of parameter l on the performance of the rsBOA on some of the used test functions is exhibited in Table 9 where it is shown that the best value of parameter l is 5.

Table 4 Computation time (minutes) comparison

	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
rsBOA	**5.1**	**5.6**	**5.0**	**5.3**	**5.3**	**5.6**	**6.3**	6.7	**6.8**	7.4	**6.9**	**7.2**	**8.3**	**9.2**
LA-rBOA	5.3	5.9	5.9	6.3	6.5	6.5	6.8	6.6	7.3	**5.4**	7.9	7.5	8.5	9.5
IrBOA	8.5	8.2	10.3	10.0	9.8	9.2	10.2	9.5	11.5	12.2	12.8	8.1	12.6	12.6
Original rBOA	6.5	6.9	6.6	6.0	6.1	6.2	6.5	**6.5**	7.0	7.2	7.1	7.5	8.5	9.3

5.2 Experiment 2: Comparison of the rsBOA with some general EDA algorithms

In this section we compare the results for the proposed rsBOA with some general EDA algorithms and one of the well-known method s for continuous global optimization (CMA-ES). To do this the proposed rsBOA is applied on the optimization functions in Table 1 in order to see how the proposed algorithm effects the optimization of the used test functions. The optimization outcomes of the proposed rsBOA are compared to five other algorithms. The five algorithms are:

- Continuous univariate marginal distribution algorithm (UMDA) [19].
- Estimation of Gaussian (Bayesian) network algorithm (EGNA) [19].
- Estimation of multivariate normal (distribution) algorithm (EMNA) [38].
- Extended compact genetic algorithm for real-parameter optimization by using adaptive discretization [38].
- A restart covariance matrix adaptation evolution strategy with increasing population size [39].

The results for the rsBOA and the other five algorithms are given in Table 10. For each function the best function values on the 50 independent runs are given. Table 10 shows that in the most used optimization functions, the rsBOA obviously

Table 5 Sensitive analysis results for parameter τ

Function	$\tau = 0.1$	$\tau = 0.3$	$\tau = 0.5$	$\tau = 0.7$
F1	**11.12E+02**	50.88E+02	88.59E+02	12.47E+03
F2	**95.88E+01**	16.91E+03	24.88E+03	70.92E+03
F3	**15.68E+06**	33.93E+06	42.38E+03	57.14E+03
F7	**16.53E+02**	16.69E+02	19.62E+02	20.86E+02
F8	**−13.91E+01**	−13.87E+01	−13.86E+01	−13.82E+01
F9	−29.37E+01	**−29.45E+01**	−29.23E+01	−29.09E+01
F10	−29.74E+01	**−30.00E+01**	−29.25E+01	−28.88E+01
F11	**99.63E+00**	99.86E+00	99.85E+00	10.05E+01

Table 6 Sensitive analysis results for parameter Np

Function	Np = 100	Np = 300	Np = 500	Np = 700	Np = 900
F1	10.71E+03	35.24E+02	26.68E+02	23.78E+02	19.92E+02
F2	18.60E+02	10.90E+02	96.48E+01	83.50E+01	76.28E+01
F3	56.38E+06	36.73E+06	15.70E+06	10.21E+06	10.70E+06
F7	18.09E+02	17.84E+02	16.85E+02	16.48E+02	16.42E+02
F8	−12.99E+01	−13.88E+01	−13.88E+01	−13.87E+01	−13.87E+01
F9	−25.65E+01	−30.05E+01	−29.43E+01	−29.38E+01	−30.37E+01
F10	−27.56E+01	−28.04E+01	−29.02E+01	−29.45E+01	−29.80E+01
F11	100.70E+00	100.10E+00	99.60E+00	99.59E+00	99.29E+00

Table 7 Sensitive analysis results for parameter *K*

Function	K = 5	K = 10	K = 20	K = 30
F1	95.2E+02	**26.68E+02**	38.15E+02	37.72E+02
F2	42.20E+02	**96.48E+01**	15.79E+02	17.94E+02
F3	86.56E+06	15.70E+06	**15.24E+06**	57.31E+06
F7	18.88E+02	16.85E+02	**16.80E+02**	18.80E+02
F8	−12.74E+01	**−13.88E+01**	−13.82E+01	−13.84E+01
F9	−25.35E+01	−29.43E+01	**−29.78E+01**	−29.12E+01
F10	−27.51E+01	**−29.02E+01**	−27.14E+01	−26.28E+01
F11	100.22E+00	**99.60E+00**	100.65E+00	99.94E+00

Table 8 Sensitive analysis results for parameter λ

Function	λ = 0.2	λ = 0.4	λ = 0.6	λ = 0.8
F1	24.33E+02	**26.68E+02**	26.78. E+02	26.93E+02
F2	**93.89E+01**	96.48E+01	94.72E+01	98.19E+01
F3	**13.96E+06**	15.70E+06	20.09E+06	21.75E+06
F7	18.25E+02	**16.85E+02**	**16.85E+02**	16.87E+02
F8	−13.80E+01	**−13.88E+01**	−13.86E+01	13.87E+01
F9	−29.13E+01	**−29.43E+01**	−28.14E+01	−29.17E+01
F10	−29.01E+01	−29.02E+01	−29.21E+01	**−29.35E+01**
F11	10.12E+01	**99.60E+00**	10.11E+01	10.14E+01

outperforms the other algorithms. Also it should be noted that CMA-ES is not really an EDA algorithm; it is an evolution strategy adapting the full covariance matrix of the normal mutation search distribution. We have considered this algorithm in the comparison because CMA-ES is a well-known method for continuous global optimization.

Table 9 Sensitive analysis results for parameter l

Function	l = 2	l = 5	l = 8
F1	23.63E+02	**26.68E+02**	28.43. E+02
F2	99.74E+01	**96.48E+01**	98.38E+01
F3	16.62E+06	**15.70E+06**	22.74E+06
F7	31.35E+02	**16.85E+02**	18.42E+02
F8	**−14.72E+01**	−13.88E+01	−13.86E+01
F9	−29.27E+01	**−29.43E+01**	−28.04E+01
F10	−29.52E+01	−29.02E+01	**−29.86E+01**
F11	10.97E+01	**99.60E+00**	10.23E+01

Table 10 Numerical results for the rsBOA and other algorithms

Function	UMDA	EGNA	EMNA	SoD ECGA	CMA-ES	rsBOA
F1	−4.40E+02	−4.43E+02	**−4.50E+02**	**−4.50E+02**	**−4.50E+02**	**−4.50E+02**
F2	−4.05E+02	−4.10E+02	−4.21E+02	**−4.50E+02**	**−4.50E+02**	**−4.50E+02**
F3	−4.04E+02	−4.15E+02	−4.21E+02	**−4.50E+02**	**−4.50E+02**	**−4.50E+02**
F4	−4.01E+02	−4.11E+02	−4.25E+02	**−4.50E+02**	**−4.50E+02**	**−4.50E+02**
F5	−2.90E+02	−2.93E+02	−3.05E+02	**−3.10E+02**	**−3.10E+02**	−3.07E+02
F6	3.50E+02	3.61E+02	3.70E+02	3.84E+02	**3.90E+02**	3.89E+02
F7	−1.65E+02	−1.70E+02	−1.72E+02	−1.80E+02	−1.80E+02	**−1.82E+02**
F8	−1.25E+02	−1.30E+02	**−1.40E+02**	−1.20E+02	−1.20E+02	**−1.40E+02**
F9	−2.90E+02	−2.95E+02	−2.97E+02	**−3.30E+02**	**−3.30E+02**	**−3.30E+02**
F10	−2.85E+02	−2.92E+02	−3.00E+02	−3.22E+02	**−3.30E+02**	−3.09E+02
F11	9.01E+01	9.45E+01	9.90E+01	**8.77E+01**	8.91E+01	9.80E+01
F12	−4.11E+02	−4.37E+02	−4.42E+02	−3.96E+02	−4.31E+02	**−4.56E+02**
F13	−1.00E+02	−1.13E+02	−1.29E+02	**−1.30E+02**	−1.29E+02	**−1.30E+02**
F14	−2.57E+02	−2.69E+02	−2.91E+02	−2.97E+02	−2.97E+02	**−2.99E+02**

6 Conclusion

This paper offers a new real-coded stochastic Bayesian optimization algorithm that uses a stochastic Bayesian network. For each potential edge in the Bayesian network there is a probability distribution function which determines the existence or lack of the corresponding edge. In order to generate each new solution, one Bayesian network is sampled according to the stochastic structure. Before the next generation is generated, each probability distribution function will be updated according to the dependence tree of the promising solutions. This dependence tree is built by the Chow-Liu algorithm. The experimental results which are reported here indicate that the proposed algorithm is superior to other related algorithms in terms of quality and performance measures.

References

1. D.E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning* (Addison-Wesley, Reading, 1989)
2. M. Gallagher, I. Wood, J. Keith, Bayesian inference in estimation of distribution algorithms, in *Proceedings of the IEEE Congress on Evolutionary Computation* (2007), pp. 127–133
3. D. Heckerman, D. Geiger, D.M. Chickering, *Learning Bayesian networks: the combination of knowledge and statistical data, Technical Report MSR-TR-94-09* (Microsoft Research, Redmond, 1995)
4. M. Pelikan, Bayesian optimization algorithm: from single level to hierarchy. Ph.D. Dissertation, University of Illinois at Urbana-Champaign, Urbana, IL (2002)
5. J.A. Lozano, P. Larrañaga, I. Inza, E. Bengoetxea (eds.), *Towards a new evolutionary computation: advances on estimation of distribution algorithms* (Springer, Berlin, 2006)
6. H. Karshenas, R. Santana, C. Bielza, P. Larrañaga, Regularized continuous estimation of distribution algorithms. Appl. Soft Comput. **13**, 2412–2432 (2013)
7. P. Larrañaga, J. Lozano (eds.), *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation* (Kluwer Academic Publishers, Norwell, 2001)
8. A. Abdollahzadeh, A. Reynolds, M. Christie, D.W. Corne, G.J. Williams, B.J. Davies, Estimation of distribution algorithms applied to history matching. SPE J **18**(03), 508–517 (2013)
9. D. Heckerman, in *Innovations in Bayesian Networks*, chap. 3, ed. by D.E. Holmes, L.C. Jain (Springer, Berlin, 2008), pp. 33–82
10. E.C. Conde, S.I. Valdez, E. Hernández, in *Multibody Mechatronic Systems*, ed. by M. Ceccarelli, E.E.H. Martinez (Springer, Berlin, 2014), pp. 315–325
11. D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization. IEEE Trans. Evol. Comput. **1**(1), 67–82 (1997)
12. C.K. Chow, C.N. Liu, Approximating discrete probability distributions with dependence trees. IEEE Trans. Inf. Theory **14**(3), 462–467 (1968)
13. B. Moradabadi, H. Beigy, C.W. Ahn, An improved real-coded Bayesian optimization algorithm for continuous global optimization. Int J Innov Comput Inf Control **9**(6), 2505–2519 (2013)
14. D.M. Chickering, D. Geiger, D. Heckerman, Learning Bayesian network is NP-hard. Technical Report MSR-TR-94-17 (1994)
15. L. Wang, Sh Wang, M. Liu, A Pareto-based estimation of distribution algorithm for the multi-objective flexible job-shop scheduling problem. Int. J. Prod. Res. **51**(12), 3574–3592 (2013)
16. M. Pelikan, K. Sastry, E. Cantu-Paz (eds.), *Scalable optimization via probabilistic modeling: from algorithms to applications* (Springer, Berlin, 2006)
17. P. Larrañaga, H. Karshenas, C. Bielza, R. Santana, A review on evolutionary algorithms in Bayesian network learning and inference tasks. Inf. Sci. **233**, 109–125 (2013)
18. C.W. Ahn, R.S. Ramakrishna, On the scalability of real-coded Bayesian optimization algorithm. IEEE Trans. Evol. Comput. **12**(3), 307–322 (2008)
19. P.A.N. Bosman, Design and application of iterated density-estimation evolutionary algorithms. Ph.D. Dissertation, Utrecht University, Utrecht, The Netherlands (2003)
20. C.W. Ahn, R.S. Ramakrishna, D.E. Goldberg, Real-coded Bayesian optimization algorithm: bringing the strength of BOA into the continuous world, in *Lecture Notes in Computer Science*, vol 3102 (Springer, Berlin, 2004), pp. 840–851
21. G. Harik, *Linkage learning in via probabilistic modeling in the ECGA* (University of Illinois at Urbana-Champaign, Urbana, 1999)
22. Function definitions and performance criteria for the special session on real-parameter optimization at CEC2005 (2005). http://www.ntu.edu.sg/home/EPNSugan
23. B. Moradabadi, H. Beigy, C.W. Ahn, An Improved real-coded bayesian optimization algorithm, in *Proceedings of IEEE Congress on Evolutionary Algorithm* (2011)
24. D.E. Goldberg, K. Sastry, *Genetic algorithms: the design of innovation*, 2nd edn. (Springer, Berlin, 2002)
25. X. Wei, Evolutionary continuous optimization by Bayesian networks and Gaussian mixture model, in *Proceedings of the IEEE 10th International Conference of Signal Processing (ICSP)* (2010), pp. 1437–1440
26. M. Pelikan, *BOA: The Bayesian Optimization Algorithm* (Springer, Berlin, 2005)

27. J. Ocenasek, J. Schwarz, Estimation of distribution algorithm for mixed continuous discrete optimization problems, in *Proceedings of the 2nd Euro-International Symposium on Computational Intelligence* (2002), pp. 227–232
28. C.W. Ahn, *Advances in Evolutionary Algorithms: Theory, Design and Practice, Studies in Computational Intelligence* (Springer, Berlin, 2006)
29. Q. Shi, S. Liang, W. Fei, Y. Shi, R. Shi, Study on Bayesian network parameters learning of power system component fault diagnosis based on particle swarm optimization. Int J Smart Grid Clean Energy **2**(1), 132–137 (2013)
30. H. Muhlenbein, T. Mahnig, FDA—a scalable evolutionary algorithm for the optimization of additively decomposed functions. Evol. Comput. **7**(4), 353–376 (1999)
31. P. Larrañaga, R. Etxeberria, J.A. Lozano, J.M. Pena, Optimization in continuous domains by learning and simulation of Gaussian networks, in *Proceedings of the 2000 Genetic and Evolutionary Computation Conference Workshop Program* (2000), pp. 201–204
32. K. Swersky, J. Snoek, R.P. Adams, in *Advances in Neural Information Processing Systems*, ed. by C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, K.Q. Weinberger (2013), pp. 2004–2012
33. B. Moradabadi, H. Beigy, C.W. Ahn, An adaptive clustering based bayesian optimization algorithm for continuous global optimization. J. Comput. Sci. Eng. **9**(1), 80–85 (2012)
34. Y.P. Chen, C.H. Chen, Enabling the extended compact genetic algorithm for real-parameter optimization by using adaptive discretization. Evol. Comput. **18**(2), 199–228 (2010)
35. A. Auger, N. Hansen, A restart CMA evolution strategy with increasing population size, in *The 2005 IEEE Congress on Evolutionary Computation*, vol 2 (2005), pp. 1769–1776
36. B. Moradabadi, H. Beigy, A new real-coded Bayesian optimization algorithm based on a team of learning automata for continuous global optimization. J. Genetic Program. Evol. Mach. **15**(2), 169–193 (2014)
37. C.H.E.N. Chao-Hong, Y.P. Chen, Quality analysis of discretization methods for estimation of distribution algorithms. IEICE Trans. Inf. Syst. **97**(5), 1312–1323 (2014)
38. S.F. Chen, B. Qian, B. Liu, R. Hu, C.S. Zhang, in *Intelligent Computing Methodologies*, ed. by D.-S. Huang, K.-H. Jo, L. Wang (Springer, Berlin, 2014), pp. 686–696
39. M. Kaedi, N.G. Aghaee, C.W. Ahn, Evolutionary optimization in dynamic environments: bringing the strengths of dynamic Bayesian networks into Bayesian optimization algorithm. Int J Innov Comput Inf Control **9**, 2485–2503 (2013)