Real-Coded Bayesian Optimization Algorithm: Bringing the Strength of BOA into the Continuous World

Chang Wook Ahn[1], R.S. Ramakrishna[1], and David E. Goldberg[2]

[1] Department of Information and Communications
Kwang-Ju Institute of Science and Technology, Gwangju 500-712, Korea
{cwan,rsr}@kjist.ac.kr
http://parallel.kjist.ac.kr/~cwan/
[2] Department of General Engieering
University of Illinois, Urbana, IL 61801, USA
deg@illigal.ge.uiuc.edu
http://www-illigal.ge.uiuc.edu/goldberg/d-goldberg.html

Abstract. This paper describes a continuous estimation of distribution algorithm (EDA) to solve decomposable, real-valued optimization problems quickly, accurately, and reliably. This is the *real-coded Bayesian optimization algorithm* (rBOA). The objective is to bring the strength of (discrete) BOA to bear upon the area of real-valued optimization. That is, the rBOA must properly decompose a problem, efficiently fit each subproblem, and effectively exploit the results so that correct linkage learning even on nonlinearity and probabilistic building-block crossover (PBBC) are performed for real-valued multivariate variables. The idea is to perform a Bayesian factorization of a mixture of probability distributions, find maximal connected subgraphs (i.e. substructures) of the Bayesian factorization graph (i.e., the structure of a probabilistic model), independently fit each substructure by a mixture distribution estimated from clustering results in the corresponding partial-string space (i.e., subspace, subproblem), and draw the offspring by an independent subspace-based sampling. Experimental results show that the rBOA finds, with a sublinear scale-up behavior for decomposable problems, a solution that is superior in quality to that found by a mixed iterative density-estimation evolutionary algorithm (mIDEA) as the problem size grows. Moreover, the rBOA generally outperforms the mIDEA on well-known benchmarks for real-valued optimization.

1 Introduction

In the community of evolutionary computation, *estimation of distribution algorithms* (EDAs), also known as *probabilistic model building genetic algorithms* (PMBGAs), have attracted due attention of late [1], [2]. Incorporating (automated) linkage learning techniques into a graphical probabilistic model, EDAs exploit a feasible probabilistic model of selected (promising) solutions found so far while efficiently traversing the search space [2]. EDAs iterate the three steps listed below, until some termination criterion is satisfied:

1. Select good candidates (i.e., solutions) from a (initially randomly generated) population (of solutions).
2. Estimate the probability distribution from the selected individuals.
3. Generate new candidates (i.e., offspring) from the estimated distribution.

It must be noted that the third step uniquely characterizes EDAs because it replaces traditional recombination and mutation operators employed by simple genetic algorithms (sGAs). Although the sGAs (with well-designed mixing operator) and EDAs deal with solutions (i.e., individuals) in quite different ways, it has been theoretically shown (and empirically observed) that their performances are quite close to each other [1], [2]. Moreover, EDAs ensure an effective mixing and reproduction of building blocks (BBs) due to their ability to accurately capture the BB structure of a given problem, thereby solving GA-hard problems with a linear or sub-quadratic performance in terms of (fitness) function evaluations (i.e., sublinear scale-up behavior) [2]-[5]. However, there is a trade-off between the accuracy of the estimated distribution and the efficiency of computation [4], [5]. For instance, a complicated, accurate model is recommended if the fitness function to be evaluated is computationally expensive.

A large number of EDAs have been proposed for discrete and real-valued (i.e., continuous) variables [1]-[6]. Depending on how intricate and involved the probabilistic models are, they are divided into three categories: *no dependencies*, *pairwise dependencies*, and *multivariate dependencies*. Among them, the category of multivariate dependencies endeavors to use general probabilistic models, thereby solving many difficult problems quickly, accurately, and reliably [2]. The more complex the probabilistic model the harder as well is the task of finding the best structure. At the expense of some computational efficiency (with regard to learning the model), they can significantly improve the overall time complexity for large decomposable problems due to their ability to largely reduce the number of (computationally expensive) fitness function evaluations [2]. *Extended compact genetic algorithm* (ecGA), *factorized distribution algorithm* (FAD), and *Bayesian optimization algorithm* (BOA) for discrete variables and *estimation of multivariate normal algorithm* (EMNA) and (*mixed*) *iterative density-estimation evolution algorithms* ((m)IDEAs) for real-valued variables belong to this category [1]-[6].

Note that the BOA is perceived to be an important effort that employs general probabilistic models for discrete variables [3], [4]. It employs techniques for modeling multivariate data by Bayesian networks so as to estimate the joint probability distribution of promising solutions. The BOA is very effective even on large decomposable (discrete) problems with tight BBs. It is only natural that the principles of BOA be tried on continuous (i.e., real-valued) variables. This attempt led to (m)IDEAs [5], [6] which exploit Bayesian Information Criterion (BIC) (that is a penalized maximum likelihood metric) for selecting a probabilistic model and employ a mixture of normal distributions for fitting the (chosen) model. Like the BOA, they do not require any problem dependent information. There is a general, but simple factorization mixture selection among the (m)IDEAs with regard to model accuracy and computational efficiency. This is called 'mIDEA' in this paper. The mIDEA clusters the selected individuals and subsequently estimates a factorized probability distribution in each cluster separately [5], [6]. It allows the mIDEA to efficiently model nonlinear dependencies by breaking up the nonlinearity between the variables and recognizing only linear relations in each cluster. This results in a better performance on epistatic and nonlinear (real-valued) problems.

It is noted that the power of BOA arises from modeling any type of dependency and realizing *probabilistic building-block crossover* (PBBC) that approximates *population-wise building-block crossover* by a probability distribution estimated from the results of proper decomposition [4]. Analogously to (one-bit) uniform crossover, the PBBC may shuffle as many superior partial solutions (i.e., BBs) as possible in order to bring about an efficient and reliable search for the optimum. However, the mIDEA cannot realize the PBBC although learning various types of dependency is possible. This is explained below.

BBs can be defined by groups of real-valued variables, each having values in some neighborhood (i.e., small interval), that break up the problem into smaller chunks which can be intermixed to reach the optimum. As the mIDEA clusters the selected individuals on the problem dimension itself, preserving and breeding BBs is quite difficult unless clusters contain many BBs at the same time. However, the probability of coming up with clusters is very small and decreases exponentially as the problem size grows. In other words, the mIDEA can hardly find an optimal solution without maintaining at least one cluster that contains most of the superior BBs. It follows that the mIDEA may not be very effective on large decomposable problems. It is noteworthy that many real-world optimization problems are bounded difficult: the problems can be (additively) decomposed into subproblems of a certain/bounded order [4], [5].

In this paper, we propose a real-coded BOA (rBOA) along the lines of BOA. The rBOA can solve various types of decomposable problems in an efficient and scalable manner, and also find a high quality solution to traditional benchmark cases.

The rest of the paper is organized as follows. Section 2 explains the rBOA in detail and Section 3 presents the experimental results obtained with the algorithms. The paper concludes with a summary in Section 4.

2 Proposed Real-Coded BOA

This section describes rBOA as a tool to efficiently solve problems of bounded difficulty with a sublinear scale-up behavior. Fig. 1 presents the pseudocode of rBOA.

```
Parameters. $\mathcal{P}$ ($\mathcal{S}$, $\mathcal{O}$): population (selected individuals, offspring), $\mathbf{Z}^i$: $i$-th subproblem
$\mathcal{S}(\mathbf{Z}^i)$ ($\mathcal{O}(\mathbf{Z}^i)$): selected individuals (offspring) that contain the genes corresponding to $\mathbf{Z}^i$,
$K$: number of mixtures, $n$: population size, $q$: offspring size, $c_i$: number of clusters on $\mathbf{Z}^i$,
$\mathbf{C}_i^j$: partial-individuals in $j$-th cluster over $\mathbf{Z}^i$.
The rest of parameters are described in the body of the paper.
Step 1. Randomly generate initial population $\mathcal{P}$
    for $i \leftarrow 0$ to $n-1$ do
        $\mathcal{P} \leftarrow$ RandomVector();
Step 2. Select $\tau$ portion individuals $\mathcal{S}$ from the population $\mathcal{P}$
    $\mathcal{S} \leftarrow$ Selection($\tau, \mathcal{P}$);
Step 3. Search a probabilistic model structure and decompose the problem from the structure
    $\zeta \leftarrow$ Search($K, \mathcal{S}$);
    $(\mathbf{Z}^0, \cdots, \mathbf{Z}^{m-1}) \leftarrow$ Decomposition($\mathbf{Y}, \zeta$)
Step 4. Fit each submodel by mixing the clustered normal distributions of subspace
    for $i \leftarrow 0$ to $m-1$ do
        $(\mathbf{C}_i, c_i) \leftarrow$ Clustering($\mathcal{S}(\mathbf{Z}^i)$);
        for $j \leftarrow 0$ to $c_i - 1$ do
            $\beta_{ij} \leftarrow |\mathbf{C}_i^j| / \sum_{k=0}^{c_i-1} |\mathbf{C}_i^k|$;
            $\boldsymbol{\theta}_j^{\mathbf{Z}^i} \leftarrow$ Estimation($\mathbf{C}_i^j$);
    $f_{(\zeta,\boldsymbol{\theta})}(\mathbf{Y}) \leftarrow \prod_{i=0}^{m-1} \sum_{j=0}^{c_i-1} \beta_{ij} f_{(\zeta^{\mathbf{Z}^i}, \boldsymbol{\theta}_j^{\mathbf{Z}^i})}(\mathbf{Z}^i)$;
Step 5. Generate offspring $\mathcal{O}$ from the joint pdf $f_{(\zeta,\boldsymbol{\theta})}(\mathbf{Y})$ on the basis of subproblems
    for $i \leftarrow 0$ to $m-1$ do
        for $k \leftarrow 0$ to $q-1$ do
            $I_c \leftarrow$ ChooseCluster($\beta_{i,0}, \cdots, \beta_{i,c_i-1}$);
            $\mathcal{O}^{\mathbf{Z}^i} \leftarrow$ Sampling($\zeta^{\mathbf{Z}^i}, f_{(\zeta^{\mathbf{Z}^i}, \boldsymbol{\theta}_{I_c}^{\mathbf{Z}^i})}(\mathbf{Z}^i)$);
Step 6. Create a new population $\mathcal{P}$ by replacing some individuals with $\mathcal{O}$
    $\mathcal{P} \leftarrow$ PartiallyReplace($\mathcal{O}$);
Step 7. If the termination criteria are not met, go to Step 2.
```

Fig. 1. Pseudocode of rBOA.

2.1 Model Selection

A *factorization* (or a *factorized probability distribution*) is a probability distribution that can be described as a product of generalized probability density functions (gpdfs) [5]. *Bayesian factorizations*, also known as *Bayesian factorized probability distributions* come under a general class of factorizations [5], [7]. A Bayesian factorization estimates a joint gpdf for multivariate (dependent) variables by a product of univariate conditional gpdfs of each random variable. The Bayesian factorization is represented by a directed acyclic graph, called a Bayesian factorization graph, in which nodes (vertices) and edges (arcs) identify the corresponding variables (in the data set) and the conditional dependencies between variables, respectively [5].

An l-dimensional real-valued optimization problem is considered. In general, a pdf is represented by a probabilistic model $\mathcal{M}$ that consists of a structure ζ and an associated vector of parameters $\boldsymbol{\theta}$ (i.e., $\mathcal{M} = (\zeta, \boldsymbol{\theta})$) [5], [6]. As the rBOA employs the Bayesian factorization, the joint pdf of a problem can be encoded as

$$f_{(\zeta,\boldsymbol{\theta})}(\mathbf{Y}) = \prod_{i=0}^{l-1} f_{\dot{\boldsymbol{\theta}}^i}(Y_i|\mathit{\Pi}_i) \tag{1}$$

where $\mathbf{Y} = (Y_0, \cdots, Y_{l-1})$ presents a vector of real-valued random variables, $\mathit{\Pi}_i$ is the set of parents of Y_i (i.e., the set of nodes from which there exists an edge to Y_i), and $f_{\dot{\boldsymbol{\theta}}^i}(Y_i|\mathit{\Pi}_i)$ is the conditional pdf of Y_i conditioned on $\mathit{\Pi}_i$ with its parameters $\dot{\boldsymbol{\theta}}^i$.

There are two basic factors behind any scheme for learning the structure of a probabilistic model (i.e., model selection): a scoring metric and a search procedure [3]-[6]. The scoring metric measures the quality of the structure of Bayesian factorization graph and the search procedure efficiently traverses the space of all feasible structures for finding the best one with regard to a given scoring metric. It may be noted that BOA and mIDEA employ a *Bayesian information criterion* (BIC) as the scoring metric and an incremental greedy algorithm as the search procedure.

Let $\boldsymbol{\mathcal{S}}$ be the set of selected individuals, viz., $\boldsymbol{\mathcal{S}} = \left(\mathbf{y}^0, \mathbf{y}^1, \cdots, \mathbf{y}^{|\boldsymbol{\mathcal{S}}|-1}\right)$. The BIC metric that should be minimized is formulated as follows:

$$\begin{aligned} BIC\left(f_{(\zeta,\boldsymbol{\theta})}(\mathbf{Y}), \boldsymbol{\mathcal{S}}\right) &= -\ln\left(\prod_{j=0}^{|\boldsymbol{\mathcal{S}}|-1} f_{(\zeta,\boldsymbol{\theta})}(\mathbf{y}^j)\right) + \lambda \ln\left(|\boldsymbol{\mathcal{S}}|\right)|\boldsymbol{\theta}| \\ &= -\sum_{j=0}^{|\mathcal{S}|-1} \ln\left(f_{(\zeta,\boldsymbol{\theta})}\left(\mathbf{y}^j\right)\right) + \lambda \ln\left(|\boldsymbol{\mathcal{S}}|\right)|\boldsymbol{\theta}| \end{aligned} \tag{2}$$

[5], [6] where λ regularizes the extent of penalty. In (2), the first and second terms represent the model fitting error and the model complexity, respectively. Since minimal negative log-likelihood is equivalent to minimal entropy, (2) is rewritten as

$$BIC\left(f_{(\zeta,\boldsymbol{\theta})}(\mathbf{Y}), \boldsymbol{\mathcal{S}}\right) = |\boldsymbol{\mathcal{S}}|\, h\left(f_{(\zeta,\boldsymbol{\theta})}(\mathbf{Y})\right) + \lambda \ln\left(|\boldsymbol{\mathcal{S}}|\right)|\boldsymbol{\theta}| \tag{3}$$

[5], [6] where $h\left(f_{(\zeta,\boldsymbol{\theta})}(\mathbf{Y})\right)$ represents the differential entropy of $f_{(\zeta,\boldsymbol{\theta})}(\mathbf{Y})$.

Although the BIC fails to exactly capture the types of interaction between variables, the important point is to have a knowledge of the variables which are dependent regardless of linearity or nonlinearity. The reason for this assertion is that the dependent type itself is learned in the model fitting phase (in Section

2.2). However, the BIC might lead to incorrect factorization if there is some kind of symmetry in the selected individuals. In other words, there is a high possibility that the dependent variables are learned as independent ones. In order to avoid this problem as well as to enhance the reliability of learning dependency, a (joint) mixture distribution is employed for modeling the selected individuals. With this in view, the BIC in (3) can be modified as (4)

$$BIC\left(K, f_{(\zeta,\boldsymbol{\theta})}(\mathbf{Y}), \boldsymbol{\mathcal{S}}\right) = \sum_{i=1}^{K} \left\{ |\boldsymbol{\mathcal{S}}_i| \, h\left(f_{(\zeta,\boldsymbol{\theta}_i)}(\mathbf{Y})\right) \right\} + K\lambda \ln\left(|\boldsymbol{\mathcal{S}}|\right) |\boldsymbol{\theta}_i| \tag{4}$$

where K is the number of mixture components, $|\boldsymbol{\mathcal{S}}_i|$ is the expected number of selected individuals drawn from a probability distribution $f_{(\zeta,\boldsymbol{\theta}_i)}(\mathbf{Y})$, and $\boldsymbol{\theta}_i$ is parameters of ith mixture component.

The incremental greedy algorithm starts with an empty graph with no edges, and proceeds by (incrementally) adding an edge that maximally improves the metric until no more improvement is possible [3]-[6]. The greedy algorithm does not find an optimal structure in general because searching for the structure is an NP-complete problem. However, the computed structure is good enough for encoding most important interactions between variables of the problem [3], [4].

A Bayesian factorization graph that represents a probabilistic model structure is obtained after factorization and application of the incremental greedy algorithm.

2.2 Model Fitting and Sampling

It may be noted that maximal connected subgraphs of a Bayesian factorization graph are the component subproblems. A hard optimization problem can thus be reduced to the problem of solving several easy subproblems (if the resulting graph consists of several maximally connected subgraphs). Any graph search algorithm can be applied to extract the maximally connected subgraphs. As the BBs in real space are defined as a set of variables with some neighborhood that can be intermixed for finding an optimum, the variables of each subproblem can eventually build up BBs in view of their close interactions.

The BOA models any type of dependency because it maintains all the conditional probabilities, without losing any information due to the finite cardinality (of the set of variables). Moreover, the BOA naturally performs the PBBC because it strictly separates and independently treats the maximally connected subgraphs all through the (probabilistic) model selection, model fitting, and (offspring) sampling phases. Hence, the BOA can solve difficult problems quickly, accurately, and reliably.

On the other hand, the mIDEA clusters the selected individuals for breaking up the nonlinear dependencies between variables and subsequently estimates a factorized probability distribution in each cluster [5], [6]. However, it cannot realize the PBBC even though any type of mixture distribution, depending on the pdf used in each cluster, can be constructed. The reason is discussed below.

The clustering is performed on the problem space itself (instead of the subproblem space) and the mixture distribution is constructed from a linear combination of factorized pdfs (estimated in clusters). In the sampling phase, an entire individual is drawn from a proportionally chosen pdf. Hence, at least one cluster must contain almost all the (superior) BBs of the problem if the aim is to find an optimal solution. In order to get such clusters, however, a huge population and a very large number of clusters are required. It may result in an exponential scale-up behavior, even if the problem is decomposable into subproblems of bounded order. In other words, the mIDEA may easily be misled by many (deceptive) suboptima as it cannot efficiently capture and boost (superior) BBs.

After finding a feasible probabilistic model through the Bayesian factorization (in Section 2.1), the promising subproblems whose variables have linear and/or nonlinear interactions are obtained by extracting maximal connected subgraphs from the resulting factorization graph. Next to the problem decomposition, a clustering is performed on each subproblem (i.e., subspace). Although any clustering algorithm can be used, a computationally inexpensive algorithm is desirable. The purpose of clustering is twofold: comprehending the nonlinearity and searching the space effectively. That is, the clustering can model the nonlinear dependencies using a combination of piecewise linear interaction models resulting from breaking up the nonlinearity. In addition, the clustering has an effect of partitioning each subspace for effective search. Thereby, the rBOA can treat each subproblem independently and then fit it efficiently by mixing pdfs even in the presence of nonlinearly dependent variables. Although Bosman [5] suggested a framework from which to carve the model as a matter of course, the focus is somewhat different from that of rBOA.

Let $\mathbf{Z}^i = \left\{Z_0^i, \cdots, Z_{|\mathbf{Z}^i|-1}^i\right\}$ be a vector of random variables of the ith subproblem (viz., $\bigcup_i \mathbf{Z}^i = \mathbf{Y}$ and $\bigcap_i \mathbf{Z}^i = \phi$), in which the variables are already topologically sorted for drawing new partial-individuals corresponding to the subproblem. Let $\zeta^{\mathbf{Z}^i}$ and $\boldsymbol{\theta}^{\mathbf{Z}^i}$ indicate a (probabilistic model) structure of the variables $\mathbf{Z}^i$ (i.e., substructure) and its associated parameters, respectively (viz., $\mathcal{M}^{\mathbf{Z}^i} = (\zeta^{\mathbf{Z}^i}, \boldsymbol{\theta}^{\mathbf{Z}^i})$), and $f_{\left(\zeta^{\mathbf{Z}^i}, \boldsymbol{\theta}_j^{\mathbf{Z}^i}\right)}\left(\mathbf{Z}^i\right)$ represent a joint pdf (i.e. probability distribution) with parameters $\boldsymbol{\theta}_j^{\mathbf{Z}^i}$ that are estimated from jth cluster over $\mathbf{Z}^i$(i.e., the ith subspace). Therefore, a joint pdf of $\mathbf{Y}$ can be constructed by a product of linear combinations of subproblem pdfs as given by

$$f_{(\zeta,\boldsymbol{\theta})}(\mathbf{Y}) = \prod_{i=0}^{m-1} \sum_{j=0}^{c_i-1} \beta_{ij} f_{\left(\zeta^{\mathbf{Z}^i}, \boldsymbol{\theta}_j^{\mathbf{Z}^i}\right)}\left(\mathbf{Z}^i\right) \tag{5}$$

where m is the number of subproblems, c_i is the number of clusters of $\mathbf{Z}^i$, β_{ij} is the mixture coefficients, $\beta_{ij} \geq 0$, and $\sum_{j=0}^{c_i-1} \beta_{ij} = 1$ for all i. In general, the mixture coefficient β_{ij} is proportional to the number of individuals of the jth cluster of $\mathbf{Z}^i$ [5], [6].

If the problem is non-decomposable, the operational mechanism of rBOA is not much different from that of mIDEA except that the mIDEA can construct a different probabilistic model for each cluster. However, the encouraging fact is that many (real-world) problems are indeed decomposable into several subproblems of bounded order [4].

As preparation for drawing new individuals (i.e., offspring), univariate conditional pdfs of the proposed mixture distribution must be derived on the basis of subproblems. Therefore, (5) is rewritten as (6) from a sampling point of view:

$$f_{(\zeta,\boldsymbol{\theta})}(\mathbf{Y}) = \prod_{i=0}^{m-1} \sum_{j=0}^{c_i-1} \beta_{ij} f_{\left(\zeta^{\mathbf{Z}^i},\boldsymbol{\theta}_j^{\mathbf{Z}^i}\right)}\left(\mathbf{Z}^i\right) = \prod_{i=0}^{m-1} \sum_{j=0}^{c_i-1} \beta_{ij} \prod_{k=0}^{\left|\mathbf{Z}^i\right|-1} f_{\ddot{\boldsymbol{\theta}}_j^k}\left(Z_k^i | \Pi_{Z_k^i}\right) \quad (6)$$

Let $\mathbf{X} = \left\{Z_k^i, \Pi_{Z_k^i}\right\}$, and $\boldsymbol{\mu}$, Σ be the mean vector and the symmetric covariance matrix of $\mathbf{X}$, respectively. Employing the normal pdf due to its inherent advantages - close approximation and simple analytic properties - the univariate conditional pdfs in (6) can be obtained from

$$f_{\ddot{\boldsymbol{\theta}}_j^k}\left(Z_k^i | \Pi_{Z_k^i}\right) = f_{\mathcal{N}}\left(X_0 | X_1, \cdots, X_{|\mathbf{X}|-1}\right) = \frac{1}{\sqrt{2\pi}\widetilde{\sigma}}\, e^{-\frac{(X_0-\widetilde{\mu})^2}{2\widetilde{\sigma}^2}} \quad (7)$$

[5], [6] where $\widetilde{\sigma} = \frac{1}{\sqrt{\left(\Sigma^{-1}\right)_{0,0}}}$, $\widetilde{\mu} = \mu_0 - \frac{\sum_{i=1}^{|\mathbf{X}|-1}(X_i-\mu_i)\left(\Sigma^{-1}\right)_{i,0}}{\left(\Sigma^{-1}\right)_{0,0}}$.

Sampling the new individuals from the resulting factorization of (6) is straightforward [5]. At first, the normal pdf over the jth cluster of the normal mixture estimate for the ith subproblem is selected with probability β_{ij}. Subsequently, a multivariate string (i.e., partial-individual) corresponding to $\mathbf{Z}^i$ can be drawn by simulating the univariate conditional pdfs (i.e., eqn. (7)) of the chosen normal pdf which models one of the promising partitions (i.e., a superior BB) of a subspace (i.e., subproblem). By repeating this for all the subproblems, superior BBs can be mixed and bred for subsequent search.

3 Experiments and Discussion

This section investigates the performance of rBOA by comparing it with that of the mIDEA through computer experiments. Solution quality returned by the fixed number of function evaluations is taken to be a performance measure. For simplicity, one normal distribution is used for Bayesian factorization (i.e., $K = 1$). The BEND leader algorithm (with a threshold value of 0.3) is used as the clustering algorithm due to its speed and flexibility [5], [6]. Truncation selection with $\tau = 0.5$ and BIC with $\lambda = 0.5$ have been invoked. Since no prior information about the problem structure is available in practice, we set $|\mathbf{Y}| - 1$ for the number of allowable parents. Each experiment is terminated when the number of function evaluations reaches $100n$, where n is the population size. All the results were averaged over 100 runs.

3.1 Test Problems

Two types of problem are considered for investigating the performance of rBOA on decomposable problems. The first test problem is a real-valued deceptive problem (RDP) composed of real-valued trap functions. The problem is given by

$$F_{RDP}(\mathbf{y}) = \sum_{i=0}^{m-1} f_{trap}\left(y_{i\cdot k}, \cdots, y_{i\cdot k+(k-1)}\right) \tag{8}$$

where $y_j \in [0,1]$, $\forall j$, k and m are the subproblem size and the number of subproblems, respectively, and $f_{trap}\left(y_{i\cdot k}, \cdots, y_{i\cdot k+(k-1)}\right)$ is a k-dimensional trap function defined by

$$f_{trap}\left(y_{i\cdot k}, \cdots, y_{i\cdot k+(k-1)}\right) = \begin{cases} 1.0, & \text{if } 0.8 \leq y_{i\cdot k+j} \leq 1.0, \forall j, \\ 0.8 - \sqrt{\frac{\sum_{j=0}^{k-1} y_{i\cdot k+j}^2}{k}}, & \text{otherwise.} \end{cases} \tag{9}$$

The second test problem is a real-valued nonlinear problem (RNP) that is constructed by concatenating Rosenbrock functions. The problem is formulated as

$$F_{RNP}(\mathbf{y}) = \sum_{i=0}^{m-1} f_R\left(y_{i\cdot k}, \cdots, y_{i\cdot k+(k-1)}\right) \tag{10}$$

where $y_j \in [-5.12, 5.12]$, $\forall j$, and $f_R\left(y_{i\cdot k}, \cdots, y_{i\cdot k+(k-1)}\right)$ is a k-dimensional Rosenbrock function defined as Table 1.

Note that the variables of the subproblem (i.e., real-valued trap function, Rosenbrock function) strongly interact with each other.

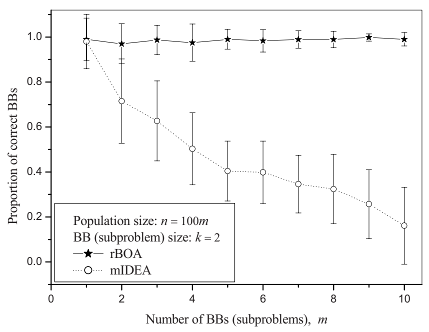
Moreover, four traditional benchmark problems that do not have any obvious ‘decomposibility’ features are also investigated. They are shown in Table 1 and should be minimized.

3.2 Experimental Results and Discussion

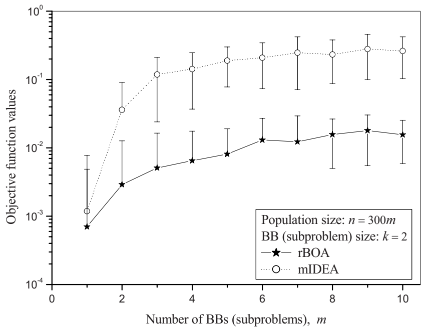
Fig. 2(a) compares the proportion of correct BBs for the algorithms as applied to the RDPs with $k = 2$ and varying m. Since a RDP consists of m subproblems, the effective problem difficulty is proportional to m in general. Hence, the population size is supplied by a linear model $\alpha \cdot m$, viz., $n = 100m$. The results show that the solution found by the rBOA is much better than that returned by the mIDEA. It is also seen that the rBOA achieves stable quality of solutions while the performance of mIDEA rapidly deteriorates as the problem size increases. That is, the rBOA exhibits a linear scale-up behavior for (additively)

Table 1. Traditional benchmark problems for numerical optimization

Problem	Function	Range
Sphere	$\sum_{j=0}^{l-1} y_j^2$	$y_j \in [-5, 5]$
Griewank	$\frac{1}{4000} \sum_{j=1}^{l-1} (y_j - 100)^2 - \prod_{j=0}^{l-1} cos\left(\frac{y_j-100}{\sqrt{j+1}}\right) + 1$	$y_j \in [-600, 600]$
Michalewicz	$\sum_{j=0}^{l-1} sin(y_j) sin^2 0 \left(\frac{(j+1)\cdot y_j^2}{\pi}\right)$	$y_j \in [0, \pi]$
Rosenbrock	$\sum_{j=1}^{l-1} \left\{100 \cdot (y_j - y_{j-1}^2)^2 + (1 - y_{j-1})^2\right\}$	$y_j \in [-5.12, 5.12]$



(a) Performance on F_{RDP} with $k = 2$ and varying m.



(b) Performance on F_{RNP} with $k = 2$ and various m.

Fig. 2. Comparison of the rBOA and mIDEA on decomposable problems.

decomposable deceptive problems; while the mIDEA has an exponential scalability. Fig. 2(b) depicts the objective function values returned by the algorithms when applied to the RNP with $k = 2$ and varying m. A linear model is also used for supplying population, i.e., $n = 300m$. It is seen that the performance of both algorithms gracefully deteriorate as the number of subproblems grows. It implies that the scale-up behavior of both algorithms becomes sublinear for decomposable nonlinear problems. However, the results show that the rBOA outperforms the mIDEA rather substantially with regard to the quality of solution. From Figs. 2(a) and (b), we may conclude that the rBOA finds a better solution with a sublinear scale-up behavior for decomposable problems than does the mIDEA. Note that the good solution and the sublinear scale-up behavior of the rBOA bring about the problem decomposition (in the model selection), and subspace-wise model fitting and (offspring) sampling operations.

Table 2 compares the solutions found by the algorithms as applied to the test functions of Table 1. The results show that the rBOA is also superior to

Table 2. Performance of the algorithms on the benchmarks ($l = 5$)

Problem Type	Population Size	mIDEA		rBOA	
		Mean	STD	Mean	STD
Sphere	$n = 500$	0.000310	0.001758	$< 10^{-7}$	-
Griewank	$n = 2000$	0.067267	0.018433	0.063001	0.016415
Michalewicz	$n = 500$	-4.606095	0.066925	-4.813710	0.019322
Rosenbrock	$n = 5000$	0.003899	0.010477	0.017825	0.091988

the mIDEA except when working on the Rosenbrock function. This is explained below.

The variables of the Rosenbrock function are highly nonlinear. In other words, they strongly interact around a curved valley. Also, it is symmetric. It is clear that incorrect factorizations (i.e., no dependencies between variables) are encountered at an early stage of rBOA and mIDEA. Due to the incorrect structure, they try to solve the problems by treating the variables in isolation. Of course, finding an optimum in this way is difficult because any given algorithm does not cross the intrinsic barrier. After a few generations, individuals start to collect round the curved valley. In fact, the rBOA can easily capture such a nonlinear symmetric dependency. However, the factorization employed one normal distribution in this experiment. It may bring about incorrect linkage learning (i.e., independent interaction) due to symmetry. This is no cause for concern when more than one normal distribution is used for factorization (the situation arising of more than one distribution is not shown in this paper, though). On the other hand, the mIDEA can cope with the cancellation effect to some extent by the use of clustering in the overall problem space.

As a result, the proposed rBOA finds a high quality solution with a sublinear scale-up behavior for decomposable problems while finding acceptable solutions to popular test problems.

4 Conclusion

This paper has presented a real-coded BOA as a continuous EDA. Decomposable problems were the prime targets. Sublinear scale-up behavior (of rBOA) was a major objective. This was achieved by linkage learning and probabilistic building-block crossover (PBBC) on real-valued variables. As a step in this direction, Bayesian factorization was performed by means of a mixture of pdfs, the substructures were extracted from the resulting Bayesian factorization graph (i.e., problem decomposition), and each substructure was fitted by mixing normal pdfs whose parameters were estimated from the subspace-based (e.g., subproblem-based) clusters. In the sampling phase, offspring were generated by a subproblem-wise sampling procedure.

Experimental studies demonstrated that the rBOA finds a better solution and exhibits a superior scale-up behavior (i.e., sublinear) (vis-a-vis the mIDEA) while encountering decomposable problems regardless of inherent problem characteristics such as deception and/or nonlinearity. Moreover, the solution of rBOA is generally better than that of mIDEA for traditional real-valued benchmarks.

Although more work needs to be done, rBOA's strategy of decomposing problems, modeling the resulting building blocks, and then searching for better solutions appears to have certain advantages over clustered model building that has been suggested and used elsewhere. Certainly, there is much work to be done in exploring the method of decomposition, the types of models utilized, as well as their computational implementation and speed, but this path appears to lead to a class of practical procedures that should find widespread use in many engineering and scientific applications.

References

1. P. Larrañaga and J. A. Lozano, *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*, Kluwer Academic Publishers, 2002.
2. M. Pelikan, D. E. Goldberg, and F. G. Lobo, "A Survey of Optimization by Building and Using Probabilistic Models," *Computational Optimization and Applications*, vol. 21, pp. 5–20, 2002.
3. M. Pelikan, D. E. Goldberg, and E. Cantú-Paz, "BOA: The Bayesian optimization algorithm," *Proceedings of GECCO'99*, pp. 525–532, 1999.
4. M. Pelikan, *Bayesian Optimization Algorithm: From Single Level to Hierarchy*, Ph. D. Thesis, University of Illinois at Urbana-Champaign, Urbana, IL, 2002.
5. P. A. N. Bosman, *Design and Application of Iterated Density-Estimation Evolutionary Algorithms*, Ph. D. Thesis, Utrecht University, TB Utrecht, The Netherlands, 2003.
6. P. A. N. Bosman and D. Thierens, "Advancing Continuous IDEAs with Mixture Distributions and Factorization Selection Metrics," *Proceedings of OBUPM workshop at GECCO'01*, pp. 208–212, 2001.
7. S. L. Lauritzen, *Graphical Models*, Clarendon Press, Oxford, 1996.