Restricted Boltzmann Machine Based Algorithm for Multi-objective Optimization

Huajin Tang, Vui Ann Shim, Kay Chen Tan and Jun Yong Chia

Abstract— **Restricted Boltzmann machine is an energy-based stochastic neural network with unsupervised learning. This network consists of a layer of hidden unit and visible unit in an undirected generative network. In this paper, restricted Boltzmann machine is modeled as estimation of distribution algorithm in the context of multi-objective optimization. The probabilities of the joint configuration over the visible and hidden units in the network are trained until the distribution over the global state reach a certain degree of thermal equilibrium. Subsequently, the probabilistic model is constructed using the energy function of the network. Moreover, the proposed algorithm incorporates clustering in phenotype space and other canonical operators. The effects on the stability of the trained network and clustering in optimization are rigorously examined. Experimental investigations are conducted to analyze the performance of the algorithm in scalable problems with high numbers of objective functions and decision variables.**

I. Introduction

EVOLUTIONARY algorithms (EAs) have been successfully implemented to solve many real-world optimization problems. Problems with multiple non-commensurable and often competing cost functions, commonly known as multi-objective optimization problems (MOOPs), are one of the main researches in EA's field [21-25]. Nonetheless, stochastic recombination from genetic operators in standard EAs may disrupt the building block of good schemas. The movement towards the optimal is, thus, extremely difficult to predict [2]. Due to these reasons, estimation of distribution algorithms (EDAs), motivated by the idea to exploit the linkage information between variables, have been introduced as a new computing paradigm in the field of evolutionary computation [8]. Neither crossover nor mutation is implemented in EDAs. Instead, the information is represented by the building of a representative probabilistic model. New solutions are subsequently generated through sampling the corresponding probabilistic model.

In the recent past, many variations of EDAs have been developed in the context of multi-objective (MO) optimization – Multi-objective estimation of distribution algorithms (MOEDAs). Mixture-based MO iterated density-estimation evolutionary Algorithm (MIDEA) [10] is one of the first EDA introduced into this field with univariate factorization as their probabilistic modeling technique. EDAs based on Bayesian network is probably the most famous algorithm proposed in MOEDA. In this approach, n-dimensional probability by product of conditional probabilities is built from Bayesian network model. The structure of the Bayesian network can be determined by detecting conditional independencies approach or score + search method, and sampling is based on Probabilistic Logic Sampling (PLS) [2], [18]. Bayesian Multi-objective optimization algorithm (BMOA) [11] and MO hierarchical Bayesian optimization algorithm (mohBOA) [13] are MOEDAs that rely on Bayesian network or particularly known as Bayesian optimization algorithm (BOA) [26]. Other well-known algorithms are MO hybrid EDA (MOHEDA) [12], MO extended compact genetic algorithm (mECGA) [14], MO Parzen-based EDA (MOPED) [15], regularity model based MOEDA (RMMEDA) [16], Decision tree based MOEDA (DT-MEDA) [17], and MO neural based EDA (MONEDA) [20].

The main characteristic that differential the EDAs is their model building technique. Most of the proposed modeling approaches in literatures used statistical inference to extract the statistical information of the selected population. However, modeling from statistical method is complicated. It may also be hard for this approach to derive accurate information from complex problems if the input data is small. Practically, real-world optimization problems are characterized by unknown properties and may have complex variables dependencies. Thus, a good modeling method should make construction easy, have low complexity, and able to model the desired distribution with good fidelity.

This paper attempts to address the issues discussed above by modeling the restricted Boltzmann machine (RBM) as estimation of distribution algorithm to solve scalable multi-objective problems. Restricted Boltzmann machine is a kind of neural network that learns the probability distribution in term of energy equilibrium. A two layered network with undirected graph in RBM is used to model the energy function of the equilibrium state. This information is captured through pair-wise interactions between visible and hidden units. The probability model is subsequently constructed, where the probability distribution is proportional to the exponential of energy value. The distributions of the solutions are clamped into marginal distribution by clamping the weighting information from all variables. This feature may help to solve the curse of dimensionality problems. Furthermore, empirical studies are carried out to investigate the performance and potential of

Huajin Tang is with the Institute for Infocomm Research, Agency for Science, Technology and Research (A*STAR), 138632, Singapore (e-mail: htang@i2r.a-star.edu.sg).

Vui Ann Shim, Kay Chen Tan, and Jun Yong Chia are with the Department of Electrical and Computer Engineering, National University of Singapore, 4 Engineering Drive 3, 117576, Singapore {e-mail: (g0800438,eletankc, g0900313)@ nus.edu.sg}.

978-1-4244-8126-2/10/$26.00 ©2010 IEEE

RBM in building the representative probabilistic model in solving scalable MOOPs. Clustering in objective space by k-mean clustering is incorporated in the algorithm. Furthermore, all algorithms are constructed with binary representation. A good model to construct the probability distribution is the one that is easy to build, sample, and return solutions with good fidelity [9]. MORBM satisfies these criteria since the statistical information is gained through a learning process and empirical results show that the algorithm is effective in solving scalable problems with high numbers of objectives and decision variables.

The remainder of this paper is organized as follow: Section II introduces the concept of restricted Boltzmann machine and its training algorithm. Section III presents the framework of the proposed algorithm, modeling, and sampling method. Test instances, implementation plus other algorithms in comparison are outlined in Section IV. Section V presents the experimental results and conclusion is drawn in section VI.

II. Restricted Boltzmann Machine

Restricted Boltzmann Machine (RBM) is a class of energy-based neural network with unsupervised learning [6], has been increasingly gaining research interests from neural network communities as feature extraction methods [3]–[5]. It is a two-layered undirected graphical model [3] where the two layers of stochastic binary neurons are the visible layer (input data) and the hidden layer (for learning capacity). The indices of visible units denoted by i , while the indices of hidden units are denoted by j. w_{ij} is the weight of the connection of the i^{th} visible unit and the j^{th} hidden unit. The structure of the RBM is illustrated in Fig 1. As can be seen, there are no connections within the same layer, and the weights are symmetric.

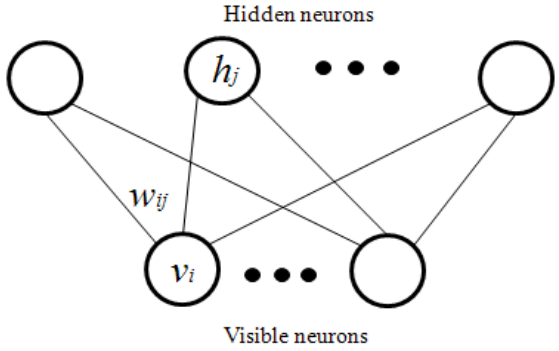

Fig. 1. Structure of RBM

The energy function of the joint configuration defined over the network is given in Eq. 1,

$$E(v,h) = -\sum_{i=1}^{n}\sum_{j=1}^{m} v_i h_j w_{ij} - \sum_{i=1}^{n} v_i b_i - \sum_{j=1}^{m} h_j b_j \quad (1)$$

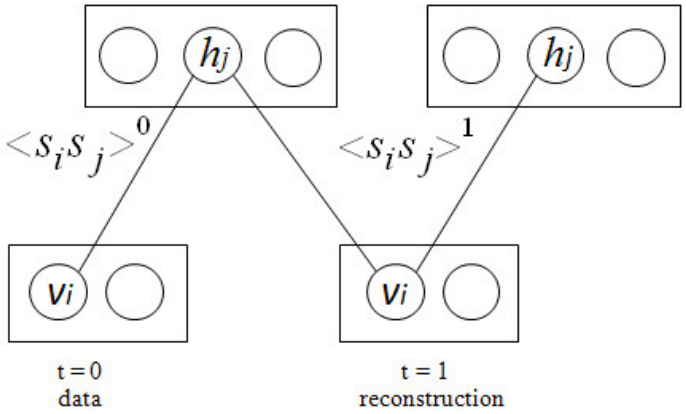

Fig. 2. RBM Training Procedure

where v_i is the state of the i^{th} visible unit, h_j is the state of the j^{th} hidden unit, b stands for bias, n and m are number of visible and hidden units. As in Hopfield net, the energy is determined by the biases and weights. The probability distribution is defined, using the energy function, as

$$p(v,h) = \frac{e^{-E(v,h)}}{Z} \quad (2)$$

where Z is the normalizing constant as shown in Eq. 3.

$$Z = \sum_{x,y} e^{-E(x,y)} \quad (3)$$

This probability (Eq. 2) is measured over both visible and hidden units and depends on the energy of the joint configuration compared with the normalizing constant. The normalizing constant is the energy of all the joint configurations. The probability of the visible unit (data point) is the sum of the probabilities of all the joint configurations that contain it, is given as follows:

$$P(v) = \sum_h P(v,h) = \frac{\sum_h e^{-E(v,h)}}{Z} \quad (4)$$

The training method for RBM, based on Constrastive Divergence (CD) [7], is crucial and is probably, the most famous and efficient training method. In order to update the weights, CD training method allow an approximation of the gradient of another objective function [6], [27] by

$$\Delta w_{ij} = \epsilon\left(< v_i h_j >_0 - < v_i h_j >_1\right) \quad (5)$$

where ϵ is the learning rate, $< v_i h_j >_0$ is the product of $v_i h_j$ before re-construction, and $< v_i h_j >_1$ is the expected value of $v_i h_j$ after one-step re-construction are clamped on the visible unit, and h_j is sampled form its posterior distribution given the reconstruction. This training method is illustrated in Fig. 2. The re-constructed samples are constructed form running the Gibbs sampler according to Eqs. 6 and 7,

$$P(v_i|h) = \varphi\left(\sum_j w_{ij} h_j - b_i\right) \quad (6)$$

$$P(h_j|v) = \varphi\left(\sum_i w_{ij} v_i - b_j\right) \quad (7)$$

where $\varphi(x) = \frac{1}{1+e^{-x}}$ is the logistic function, b_i is the bias for visible unit and b_j is the bias for hidden unit, $P(v_i|h)$ is the probability returned for v_i given h and $P(h_j|v)$ is the conditional probability of h_j given v.

III. Proposed Algorithm

A. *Basic Idea*

EDAs differ from standard genetic algorithms (GAs) due to the absence of operators, such as crossover and mutation. These operators are replaced by a representative probabilistic model of the selected population. RBM, an energy-based network, is suitable to model as EDA since the probability of the joint configuration over the visible and hidden layer is proportional to the energy of the joint configuration as given in Eq. 8.

$$p(v,h) \propto e^{-E(v,h)} \quad (8)$$

In the RBM network, the dependencies between the variables are learnt through the learning process and the information is stored as connection weights and biases. This feature is expected to steer towards Pareto optimal front since linkage information is considered in modeling the probability distribution.

B. *Algorithm Framework*

The proposed algorithm for multi-objective optimization called multi-objective restricted Boltzmann machine (MORBM) can adapt any operators (ranking, selection, archiving) in standard MOEA by replacing the crossover and mutation with RBM. In this implementation, non-dominance ranking, crowding distance, and binary tournament selection, similar to the operators in NSGAII [19], are utilized. Moreover, clustering based on k-mean clustering is employed. The overall algorithm works as follows:

Step (1) Initialization: At generation g=0; randomly generate N initial population $Pop(0)$.

Step (2) Ranking: Perform non-dominated ranking and crowding distance over the population.

Step (3) Selection: Select N subpopulation (SUB_g^{Se}) based on binary tournament selection.

Step (4) Clustering: Divide the selected population (according to objective value) into k cluster based on k-mean clustering.

Step (5) Modeling: In each cluster, build an isolated network (k network) based on RBM and perform CD training (Eq. 5).

Step (6) Reproduction: Compute probability of the joint configuration $P(v)$ (Eq. 4). Generate a new set of N solutions (P) from $P(v)$.

Step (7) Archiving: Select N individuals form $P \cup Pop\ (g)$ to form $Pop(g+1)$.

Step (8) Stopping Criteria: If stopping condition is reached, terminate. Else g = g + 1, and go to (2).

C. *Modeling*

In this implementation, each variable in the cost function is the visible unit in the network. Therefore, N variables in fitness functions mean N visible units in RBM. The function of hidden neurons is to learn the significant dependency between the variables [3] and the setting of the number of hidden unit is will be investigated. The complexity of the network would increase with the number of visible and hidden units. Since modeling will be conducted at every generation, it is essential for models to be kept simple. Therefore, number of hidden unit is set to as small as possible as long as the probability is representative. An extensive modeling is unnecessary since the exact distribution of the selected population does not represent the distribution for optimal solutions. Clustering is incorporated into the algorithm since the decomposition of a problem into sub problems has been proven to improve the optimization [10], [13]. For sake of simplicity, k is determined by user. In the implementation, a RBM is built for each cluster, $\mathcal{L}^1, \mathcal{L}^2, \ldots, \mathcal{L}^k$. Each cluster $\mathcal{L}$ undergoes two phases (positive and negative) to train the network before the weights and biases are updated. During the positive phase, the conditional probability of hidden unit ($P(h_j|v)_o$) is computed based on the input data given by Eq. 7. Based on this probability, the state of hidden neuron $< h_j >_0$ is sampled. $<>_0$ denotes the state of the neurons before reconstruction, and $<>_1$ denotes the state of the neurons after a one step reconstruction. During the negative phase, the hidden layer is stochastically fired to reconstruct the visible layer by sampling according to Eq. 6. The hidden layer is, subsequently, recomputed from the visible layer using Eq. 7. The overall training procedure is summarized below:

Step (1): $< v_i >_0 \leftarrow \mathcal{D}$; $\mathcal{D}$ is input data; epoch = 0;

Step (2): Compute $P(h_j|v)_o$ from Eq. 7

Step (3): $< h_j >_0 \sim P(h_j|v)_o$; sample $< h_j >_0$ from $P(h_j|v)_o$

Step (4): Compute $< v_i >_1$ by sampling according to Eq. 6

Step (5): Compute $< h_j >_1$ by sampling according to Eq. 7

Step (6): Update weights and biases

$$w_{ij}' = w_{ij} + \epsilon\left(< v_i h_j >_0 - < v_i h_j >_1\right) \quad (9)$$

$$b_i' = b_i + \epsilon\left(< v_i >_0 - < v_i >_1\right) \quad (10)$$

$$b_j' = b_j + \epsilon\left(< h_j >_0 - < h_j >_1\right) \quad (11)$$

Step (7): If stopping condition is reached, terminate. Else epoch = epoch + 1, and go to step (2).

D. *Reproduction*

The new population $Pop(g+1)$ is generated by sampling the probability obtained from each cluster $\mathcal{L}$. In this implementation, N new solutions are generated and equal numbers of individuals are sampled from each cluster. In binary representation, joint probability distribution with n variables in g generation is formularized as:

$$P_g(v) = p\left(v|SUB_{g-1}^{Se}\right) = \prod_{i=1}^{n} p_g(v_i) \quad (12)$$

TABLE I
TEST INSTANCES

Test Function	Objective Functions	Characteristics
ZDT1	$f_1(x) = x_1$ $f_2(x) = \mathrm{g(x)}(1 - \sqrt{f_1(x)/g(x)})$ $g(x) = 1 + 9(\sum_{i=2}^{n} x_i)/(n-1)$	Convex Pareto front n = 100 or Scalable $x_i \in [0,1]$
ZDT2	$f_1(x) = x_1$ $f_2(x) = g(x)(1 - (f_1(x)/g(x))^2)$ $g(x) = 1 + 9(\sum_{i=2}^{n} x_i)/(n-1)$	Non-convex Pareto front n = 100 or Scalable $x_i \in [0,1]$
ZDT3	$f_1(x) = x_1$ $f_2(x) = g(x)(1 - \sqrt{f_1(x)/g(x)} - f_1(x)/g(x)\ sin(10\Pi f_1(x)))$ $g(x) = 1 + 9(\sum_{i=2}^{n} x_i)/(n-1)$	Non-continuous convex Pareto front n = 100 or Scalable $x_i \in [0,1]$
DTLZ2	$f_1(x) = \big(1 + g(x)\big)\cos(x_1\,\pi/2) \ldots \cos(x_{M-1}\,\pi/2)$ $f_2(x) = \big(1 + g(x)\big)\cos(x_1\,\pi/2) \ldots \sin(x_{M-1}\,\pi/2)$ $\vdots$ $f_M(x) = \big(1 + g(x)\big)\sin(x_1\,\pi/2),$ $0 \le x_i \le 1, \text{for } i = 1,2,\ldots,n,$ where $g(x) = \sum_{x_i \in x_M}(x_i - 0.5)^2$	Spherical Pareto Optimal front n = M + k – 1, where k=10 scalable in variables number scalable in objectives number $x_i \in [0,1]$
DTLZ3	$f_1(x) = \big(1 + g(x)\big)\cos(x_1\,\pi/2) \ldots \cos(x_{M-1}\,\pi/2)$ $f_2(x) = \big(1 + g(x)\big)\cos(x_1\,\pi/2) \ldots \sin(x_{M-1}\,\pi/2)$ $\vdots$ $f_M(x) = \big(1 + g(x)\big)\sin(x_1\,\pi/2),$ $0 \le x_i \le 1, \text{for } i = 1,2,\ldots,n,$ where $g(x) =$ $100\,\big(\lvert X_M\rvert + \sum_{x_i \in X_M}(x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5))\big)$	Many local Pareto Optimal Front n = M + k – 1, where k=10 or scalable in variables number scalable in objectives number $x_i \in [0,1]$

The marginal probability of each variable is obtained through Eq. 4. Expanding the equation,

$$P(v^+) = \sum_{h=1}^{m} e^{-E(v=1,h)} \tag{13}$$

$$P(v^-) = \sum_{h=1}^{m} e^{-E(v=0,h)} \tag{14}$$

where $P(v^+)$ is the marginal cost of v_i when the cardinality of $v_i = 1$, while $P(v^-)$ is the marginal cost of v_i when the cardinality of $v_i = 0$. Therefore, the marginal probability when the cardinality of $v_i = 1$ is shown as below:

$$P(v_i = 1) = \frac{P(v^+)}{P(v^+) + P(v^-)} \tag{15}$$

Direct sampling from Eq. 15 reaches a limit in progress when the probability reaches maximum (1.0) or minimum (0.0). Therefore, the lower and upper bound are added to the probability distribution based on average cost of cardinality ($avg(P)$). The modified version of marginal probability is given as below:

$$P(v_i = 1) = \frac{P(v^+) + avg(P(v))}{P(v^+) + P(v^-) + r_i * avg(P(v))} \tag{16}$$

where r_i is the number of different values that $\mathrm{v_i}$ may take. In binary case, r_i is 2. Basing on the probability, the offspring is generated by sampling the probability according to Eq. 17.

$$x_i = \begin{cases} 1 & \text{if } random\ (0,1) \le P(v_i = 1) \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

IV. Test Instances and Implementation

This section presents the test functions which are used to evaluate the performance of the algorithm. Two other algorithms in comparison are also outlined. Finally, the experimental setting is presented.

A. *Test Functions*

Five test instances (ZDT1, ZDT2, ZDT3, DTLZ2, and DTLZ3) are used to test the performance of MORBM with scalability in the number of decision variables and objective functions. ZDT1, ZDT2, and ZDT3 have two objective functions and scalable numbers of decision variables. They are characterized by convex, non-convex, and non-continuous convex Pareto optimal front respectively. DTLZ2 and DTLZ3 are scalable problems in terms of the number of objectives and variables. The definition and characteristics of the test instances are summarized in Table 1.

B. *Other Algorithm in Comparison*

1) NSGAII

NSGAII [19] is one of the most famous algorithms in multi-objective evolutionary algorithm due to its ability to achieve promising solutions for most of the MOOPs. This algorithm uses Pareto ranking and crowding distance as fitness assignment operators, selection based on binary tournament, uniform crossover, bit-flip mutation, and parent-offspring archiving. Clustering based on k-mean clustering is incorporated into the algorithm for fair comparison with other algorithms.

2) MOUMDA

The basic architecture of multi-objective univariate marginal distribution algorithm (MOUMDA) is very similar to the MIDEA proposed by Thierens and Bosman [10]. One difference from MIDEA, is that clustering is based on k-mean clustering instead of leader algorithm. UMDA models the distribution of the selected population without considering any dependency between the variables. The basic framework of MORBM is similar to this algorithm except that linkage information is taken into consideration.

C. *Implementation*

A comparative study of MORBM, MOUMDA and NSGAII is carried out to examine their performance in problems with high number of decision variables and objective functions. All algorithms are implemented in C++ coding and ran on an Intel Pentium 4, 3.0 GHz personal computer. The experimental settings are as follows.

- Population Sizes: 100 for ZDT problems, 200 for DTLZ problems with 3 and 5 objectives, while 1500 for 7 objectives.
- Hidden units in MORBM: 20 for ZDT problems and 5 for DTLZ problems
- Training epochs in MORBM: 10 for all test instances
- Learning rate in MORBM: 0.1
- Number of clusters: 1 in ZDT and DTLZ with 3 objectives, 7 for DTLZ with 5 objectives, and 14 for DTLZ with 7 objectives.
- Stopping criteria: 200 generations for ZDT problems and 400 generations for DTLZ problems
- Mutation in NSGAII: 1/(Variable_size * Variable_bit)
- Crossover in NSGAII: 0.8
- Bits per variable: 10 bits
- Independent runs: 10 runs
- Decision variables: 100 for ZDT and n=M+K-1 for DTLZ, where M is objective number and K=10.

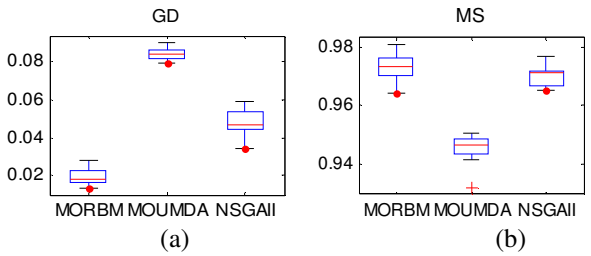

Fig. 3. Performance metric of (a) GD and (b) MS for ZDT1

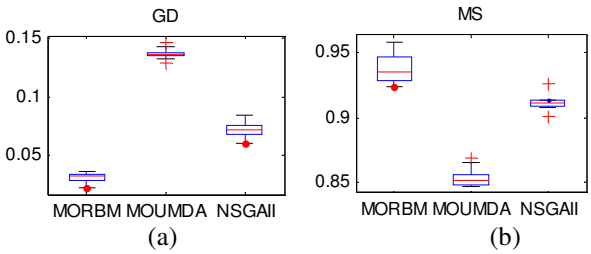

Fig. 4. Performance metric of (a) GD and (b) MS for ZDT2

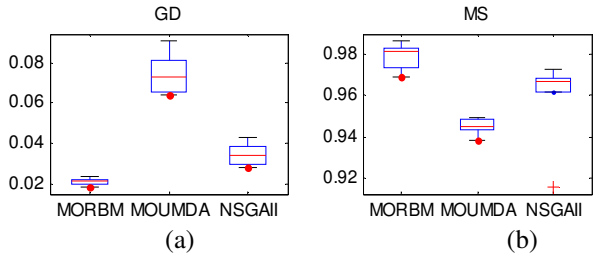

Fig. 5. Performance metric of (a) GD and (b) MS for ZDT3

V. Simulation Results

The results are presented through General Distance (GD) and Maximum spread (MS) indicators. GD measures the proximity of the simulated solutions (PF^*) compared to Pareto optimal Front (PF). A lower value indicates better performance. Meanwhile, MS measure the diversity between PF^* and PF, and higher value indicates better performance.

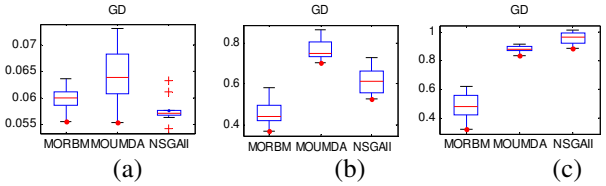

Fig. 6. Performance metric of GD for DTLZ2 with (a) 3 objectives, (b) 5 objectives, and (c) 7 objectives

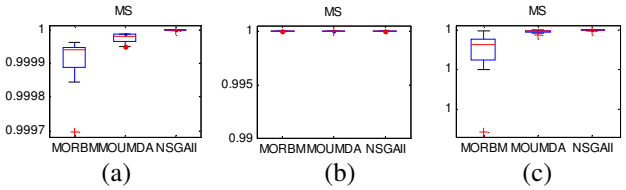

Fig. 7. Performance metric of MS for DTLZ2 with (a) 3 objectives, (b) 5 objectives, and (c) 7 objectives

A. *ZDT problems:*

Figs. 3-5 show the performance metrics of (a) GD and (b) MS for ZDT1, ZDT2, and ZDT3, respectively. It is clear that MORBM outperform NSGAII and MOUMDA on all the test instances in term of the proximity and diversity. There may have certain implicit relationship between decision variables which is hard to measure and predict. The incorporation of dependency information to predict the true probability distribution allows MORBM to obtain better results as compared to other two algorithms in comparison.

B. *DTLZ problems:*

DTLZ2 is characterized by spherical Pareto optimal front and DTLZ3 is characterized by multi-modality. In these problems, the number of objectives will be scaled up to challenge the algorithms in driving the search towards high-dimensional Pareto front. Figs. 6-7 show the performance metrics of GD and MS for DTLZ2 with (a) 3

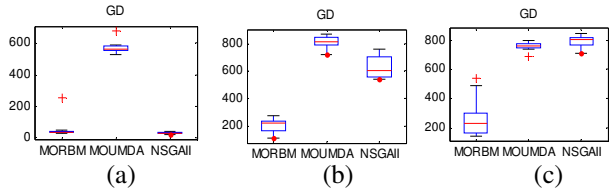

Fig. 8. Performance metric of GD for DTLZ3 with (a) 3 objectives, (b) 5 objectives, and (c) 7 objectives

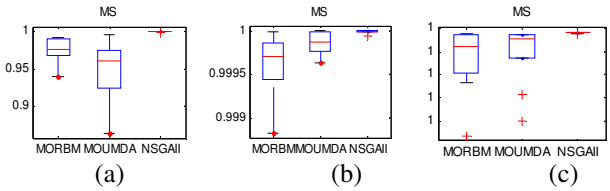

Fig. 9. Performance metric of MS for DTLZ3 with (a) 3 objectives, (b) 5 objectives, and (c) 7 objectives

objectives, (b) 5 objectives, and (c) 7 objectives. There is not much difference in performances for problems with 3, 5, and 7 objectives in terms of MS. This which implies all algorithms spread well in this problem. In term of GD, NSGAII performs slightly better than MORBM for three objectives problem while MORBM outperforms other algorithms for problem with higher objectives. The same results are observed for DTLZ3 as presented in Figs 8-9. Therefore, it can be concluded that MORBM maintained good performances with different number of objective function than other two algorithms in comparison.

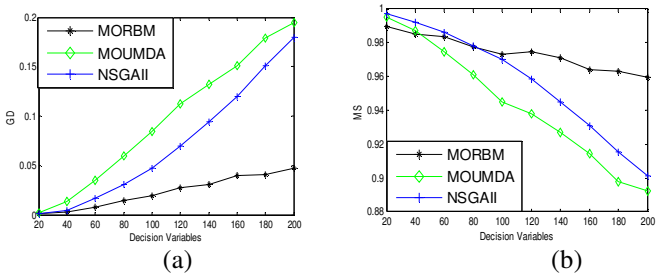

Fig. 10. Performance metric of (a) GD and (b) MS versus the number of decision variables for ZDT1.

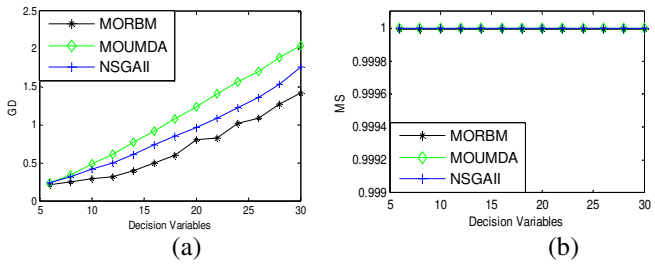

Fig. 11. Performance metric of (a) GD and (b) MS versus the number of decision variables for DTLZ2

C. *Scalable in decision variables*

The challenge of the problems increased proportionally with the number of decision variables. In order to observe the performance of various algorithms against different decision variables, Fig. 10 plots the (a) GD and (b) MS for ZDT1 from 20 up to 200 decision variables, while the performance metrics for DTLZ2 with 3 objectives from 6 up to 30 decision variables is illustrated in Fig. 11. From the figure, it is clear that MORBM performs significantly better than NSGAII and MOUMDA for ZDT1 with higher decision variables in terms of both proximity and diversity preservation. For DTLZ2, MORBM also shows better performance in GD as compared to other two algorithms. Since same population size, clustering, and selection operator are incorporated in all algorithms, therefore we can claim that MORBM performs better for higher number of decision variables than other two algorithms.

D. *Further Analyses*

This section carries out further investigations on the effect of population size, number of hidden unit, number of training epoch, and number of cluster in MORBM. ZDT1 and DTLZ2 with 5 objectives are used in these investigations.

1) Effect of population sizing

Simulations are conducted for 20, 50, 100, 200, and 500 population sizes, and the results are tabulated in Table II. All simulations stop at same fitness evaluations as stated in section IV. According to the table, NSGAII prefers larger generation rather than larger population size for better performance, while MORBM needs larger population size rather than lager number of generations for better performance, in two objective problems. This may be due to the fact that MORBM can more accurately model the distribution of the solutions when the input data is large. For problem with 5 objectives, performance is better with the increasing of population size. In this problem, many of the solutions are non-dominated, this may reduce the selection pressure to select fitter individuals. The increase of population size could produce more fit individuals.

2) Effect of Hidden unit

The performance of MORBM with different number of hidden unit is presented in Table III. From the table, it is observed that small number of hidden unit (1 unit) and large number of hidden unit (50 units) give worse performance than intermediate number of hidden unit (5-20 units). The function of hidden neuron is for learning capacity which determines the probabilities of the joint configuration over the network. Even through large number of hidden unit may reduce the final training error, but the empirical results show that intermediate number of hidden unit give better performance. This observation suggests that extensively modeling of the distribution for a selected population at particular generation is unnecessary. On the other hand, having too few hidden neuron may yield large training error which may deviate from the true distribution.

3) Effect of Training epoch

The training error in the network particularly corresponds to the number of hidden unit and training epoch. Thus, training epoch may affect the progress of

TABLE II
PROXIMITY METRIC FOR ZDT1 AND DTLZ2 WITH DIFFERENT POPULATION SIZE

	MORBM						NSGAII					
	ZDT1-100			DTLZ2-5-14			ZDT1-100			DTLZ2-5-14		
Pop Size	Mean	Std Dev	Median	Mean	Std Dev	Median	Mean	Std Dev	Median	Mean	Std Dev	Median
20	0.3034	0.0186	0.3035	1.0266	0.1320	1.0409	**0.0282**	0.0088	**0.0301**	0.6349	0.2037	0.6341
50	0.0524	0.0058	0.0536	1.0817	0.1307	1.0515	0.0460	0.0067	0.0473	0.9072	0.2184	0.7926
100	0.0194	0.0046	0.0183	1.2677	0.1779	1.3075	0.0476	0.0070	0.0464	0.8820	0.1307	0.8539
200	**0.0105**	**0.0023**	**0.0105**	0.3956	0.0736	0.3771	0.0641	**0.0054**	0.0638	0.6126	0.0663	0.6126
500	0.0110	0.0069	0.0180	**0.2211**	**0.0320**	**0.2186**	0.3043	0.0078	0.3060	**0.5024**	**0.0423**	**0.5105**

TABLE III
PROXIMITY METRIC FOR ZDT1 AND DTLZ2 WITH DIFFERENT NUMBER OF HIDDEN UNIT

	MORBM					
	ZDT1-100			DTLZ2-5-14		
Hidden Unit	Mean	Std Dev	Median	Mean	Std Dev	Median
1	0.4350	0.0377	0.4320	0.4715	0.1269	0.4180
5	0.1661	0.0105	0.1649	**0.3956**	0.0736	**0.3771**
10	0.0531	0.0086	0.0503	0.4559	0.0678	0.4372
20	**0.0194**	**0.0046**	**0.0183**	0.5453	0.1458	0.5619
50	0.0306	0.005	0.0288	0.7545	**0.0645**	0.7614

TABLE IV
PROXIMITY METRIC FOR ZDT1 AND DTLZ2 WITH DIFFERENT NUMBER OF TRAINING EPOCH

	MORBM					
	ZDT1-100			DTLZ2-5-14		
Epoch	Mean	Std Dev	Median	Mean	Std Dev	Median
1	0.0776	0.0037	0.0776	0.7436	**0.0375**	0.7450
5	0.0693	0.0048	0.0611	0.5258	0.0974	0.4914
10	0.0194	0.0046	0.0183	0.3956	0.0736	**0.3771**
20	0.0158	0.0032	0.0157	**0.3882**	0.0689	0.3799
50	0.0149	0.0035	0.0143	0.4651	0.1536	0.3968
100	**0.0135**	**0.0022**	**0.0134**	0.5381	0.0942	0.5487
500	0.0164	0.0038	0.0153	0.6052	0.1633	0.5600

TABLE V
PROXIMITY METRIC FOR ZDT1 AND DTLZ2 WITH DIFFERENT NUMBER OF CLUSTER

	MORBM					
	ZDT1-100			DTLZ2-5-14		
Cluster	Mean	Std Dev	Median	Mean	Std Dev	Median
1	**0.0194**	**0.0046**	**0.0183**	1.0815	0.1879	1.1029
3	0.0532	0.0083	0.0525	0.6810	0.1767	0.7362
5	0.1131	0.0100	0.1115	0.5372	0.2217	0.4868
7	0.1502	0.0215	0.1518	**0.3956**	**0.0736**	**0.3771**
9	0.2334	0.0280	0.1867	0.4910	0.0870	0.4827
11	0.2651	0.0458	0.2632	0.5800	0.1346	0.5782
13	0.4358	0.0485	0.4413	0.6410	0.0886	0.6280
15	0.5080	0.0383	0.5019	0.7383	0.1392	0.7307

evolution. Table IV shows the proximity metric for ZDT1 and DTLZ2 with different number of training epoch. Base on the table, training epochs that are too small or too large may deteriorate the performance. 10-20 training epochs give better performance. This result suggests that extensive training is unnecessary since the selected population may not represent the overall optimal distribution. A well defined distribution can provide the flexibility to explore area unexplored by in the previous population.

4) Effect of clustering

The effect of different number of cluster is studied and the result is tabulated in Table V. According to the table, clustering is unnecessary for ZDT1, but is of utmost importance to DTLZ2. Clustering is important for problems with solutions which are hard to represent by a single distribution. Since the coverage solutions of ZDT1 are quite evenly distributed, the clustering would only reduce the number of data for modeling. Clustering is importance for high-dimensional problems with complex Pareto optimal front to decompose the representation solutions into different region for modeling purpose. Therefore, this paper concludes that the necessity of clustering is problem dependent.

VI. CONCLUSION

In this paper, we present a new EDA based on RBM in the context of multi-objective optimization. Unlike most of the EDAs which model the probability by statistical method, MORBM models the probability through unsupervised learning and stores the statistical information in the network's weights and biases. The probability distribution of the population is proportional to the energy function of the network. Learning characteristic in MORBM may give flexibility to the algorithm in models hard and complex distribution. The k-mean clustering is employed to cluster the population, in objective space, into smaller clusters, which has been shown to improve the performance for problems with high dimensional Pareto front. New solutions are then sampled from the model thus built. The experimental studies show that MORBM scale well as compared to NSGAII and MOUMDA in the number of objective functions and decision variables. The effects of population size, number of hidden unit, number of training epoch, and number of cluster in the evolutionary process have been experimentally studied. More research should carry out to study the model building and sampling approach in order to handling problems with complicated Pareto front.

ACKNOWLEDGEMENT

This research was supported by Academic Research Fund (AcRF) under Project No.: R-263-000-480-112.

REFERENCES

[1] C. K. Goh, Y. S. Ong, and K. C. Tan, "An Investigation on Evolutionary Gradient Search for Multi- objective Optimization," *Proc. of 2008 IEEE Congress on Evolutionary Computation*, pp. 3742-3747, 2008.

[2] P. Larrañaga and J. A. Lozano, "*Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*," Norwell, MA: Kluwer, 2001.

[3] R. Salakhutdinov, A. Mnih, and G. Hinton, " Restricted Boltzmann machines for collaborative filtering", *Proceedings of the 24th international conference on Machine learning*, pp. 791–798, 2007.

[4] P. Gehler, A. Holub, and M. Welling, "The rate adapting poisson model for information retrieval and object recognition," *Proceedings of the 23rd International Conference on Machine Learning*, pp.337–344, 2006.

[5] G. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks,"*Science*, vol. 313, no. 5786, pp. 504 – 507, 2006.

[6] G. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, Vol. 14, no. 8, pp 1771-1800, 2006.

[7] T. Tieleman, "Training restricted Boltzmann machines using approximations to the likelihood Gradient," *Proceedings of the 25th Annual International Conference on Machine Learning*, pp.1064–1071, 2008.

[8] H. Muhlenbein and G. Paaß, "From recombination of genes to the estimation of distribution algorithm I. Binary parameters," *Proceedings of the 4th International Conference on Parallel Problem Solving from Nature*, pp. 178-187, 1996.

[9] Q. Zhang, "On stability of fixed points of limit models of univariate marginal distribution algorithm and factorized distribution algorithm," *IEEE Transactions on evolutionary computation*, vol. 8, no. 1, pp. 80-93, 2004.

[10] D. Thierens and P. A. N. Bosman, "Multi–objective mixture–based iterated density estimation evolutionary algorithms," *Proceedings of the GECCO–2001 Genetic and Evolutionary Computation Conference*, pp. 663–670, 2001.

[11] M. Laumanns and J. Ocenasek, "Bayesian Optimization Algorithms for Multi-Objective Optimization," *Proceedings of the 7th International Conference on Parallel Problem Solving from Nature*, pp. 298-307, 2002.

[12] Q. Zhang, J. Sun, E. P. K. Tsang, and J. A. Ford, "Hybrid Estimation of Distribution Algorithm for Global Optimisation," *Engineering computations*, vol. 21, pp. 91-107, 2004.

[13] M. Pelikan, K. Sastry, and D. Goldberg, "Multiobjective hBOA, clustering, and scalability," *Proceedings of the 2005 conference on Genetic and evolutionary computation*, pp. 663-670, 2005.

[14] K. Sastry, D. Goldberg, and M. Pelikan, "Limits of scalability of multiobjective estimation of distribution algorithms," *IEEE Congress on Evolutionary Computation*, 2005, vol. 3, pp. 2217–2224, 2005.

[15] M. Costa and E. Minisci, "MOPED: A Multi-objective Parzen-based Estimation of Distribution Algorithm for Continuous Problems," *Proceedings of the Second International Conference on Evolutionary Multi-Criterion Optimization*, page 282-294, 2003.

[16] Q. Zhang, A. Zhou, and Y. Jin, "RM-MEDA: A regularity model based multiobjective estimation of distribution algorithm," *IEEE Transaction on Evolutionary Computation*, vol. 12, pp. 41-63, 2008.

[17] X. Zhong and W. Li, "A decision-tree-based multiobjective estimation of distribution algorithm," *International Conference on Computational Intelligence and Security*, pp. 114-118, 2007.

[18] S. K. Shakya, "*DEUM : A framework for an Estimation of Distribution Algorithm based on Markov Random Fields*," PhD thesis, Robert Gordon University, Aberdeen, Scotland, 2006.

[19] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGAII," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182-197, 2002.

[20] M. Luis, G. Jesús, B. Antonio, and M. M. José, "Solving Complex High-Dimensional Problems with the Multi-objective Neural Estimation of Distribution Algorithm," *Proceedings of the 11th Annual conference on Genetic and evolutionary computation,* pp. 619-626, 2009.

[21] K. Deb, "*Multi-objective Optimization using Evolutionary Algorithms*," John Wiley & Sons, Chichester ,2001.

[22] K. C. Tan, E. F. Khor and T. H. Lee, "*Multiobjective Evolutionary Algorithms and Applications*," Springer-Verlag, 2005.

[23] C. A. Coello Coello, "Evolutionary multi-objective optimization: a historical view of the field," *Computational Intelligence Magazine, IEEE* , vol.1, no.1, pp. 28-36, Feb. 2006

[24] C. A. Coello Coello, "The EMOO repository: a resource for doing research in evolutionary multiobjective optimization," *Computational Intelligence Magazine, IEEE* , vol.1, no.1, pp. 37-45, Feb. 2006

[25] J. H. Kim and C. H. Lee, "Multi-objective evolutionary generation process for specific personalities of artificial creature," *Computational Intelligence Magazine, IEEE* , vol.3, no.1, pp.43-53, February 2008

[26] M. Pelikan, D. Goldberg, and E. Cantu-Paz, "BOA: The Bayesian Optimization Algorithm," *Proceedings of the Genetic and Evolutionary Computation Conference GECCO99*, pp. 525-532, 1999.

[27] G. Hinton, "What kind of a graphical model is the brain?" *Proceedings of the 19th international joint conference on Artificial intelligence*, pp. 1765-1775, 2005.