Chaos Elitism Estimation of Distribution Algorithm

Qingyang Xu

***Abstract*—Estimation of distribution algorithm (EDA) is a kind of EAs, which is based on the technique of probabilistic model and sampling. This paper presents a chaos elitism EDA to improve the performance of traditional EDA to solve high dimensional optimization problems. The famous elitism strategy is introduced to maintain a good convergent performance. The chaos perturbation strategy is used to improve the local search ability. Some simulation experiments conducted to verify the performance of CEEDA. The results of CEEDA are promising, and it is comparable with other EDA.**

I. Introduction

In recent years, the Estimation of Distribution Algorithm(EDA) has attracted a lot of attention. It was proposed by Miuhlenbein and Paaß [1], and emerged as a generalization of EAs, for overcoming intrinsic disadvantages of EAs, like building blocks broken and poor performance in high dimensional problems and the difficulty of modeling the solution distribution. Compared with blocks building in EAs, EDA has some attractive characteristics. It does not use the recombination or mutation operators. Instead, they extract the global statistical information from the superiority individual and build the probability model of solution distribution[2]. It is the main advantages of EDA over EAs that the explanatory and transparency of the probabilistic model guides the search process [3]. The new solutions come from the sampling of established probability model which approximates the distribution of promising solutions [4]. Such reproduction procedure allows EDA to search for the global optimal solutions effectively. Additionally, the priori information about the problem structure can be captured by the probability model estimated during the search [5].

For large-scale problem, the optimization results of EDAs become unreliable [6] , especially for the increases of number of variables and the number of mixture components. Additionally, the computational cost is huge considering all the possible (in) dependencies among the variables [7]. Therefore, in this paper we adopt a univariate Gaussian model to approximate the solution distribution. In the Gaussian model, some parameters are learnable. In this paper, we propose an chaos elitism EDA for large scale optimization problem. The learning rate of Gaussian parameter is adaptive in the optimization process. The elitism strategy is used to enhance the convergent performance, which is a popular strategy in EAs. In order to improve the local search ability, a chaos perturbation operator is designed. The local search operator enhances the diversity of population in the iteration.

Manuscript received May 12, 2014. This work was supported in part by the National Natural Science Foundation of China under Grant 61174044.

Qingyang Xu is with the Shandong University, Weihai, 264209 China (86-0631-5688338; fax: 86-0631-5688338; e-mail: xuqy1981@163.com).

II. Chaos Elitism Estimation Of Distribution Algorithm

Estimation of distribution algorithm is a series of EAs based on probability theory, which makes use of estimation and sampling technology to approximate solutions distribution and generate new solutions. The figure 1 is the flowchart of EDA.

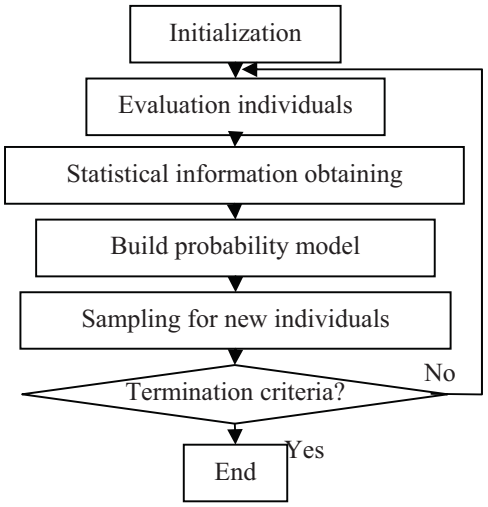

Fig. 1. Flowchart of EDA

A. The probability model build and updating mechanism

The most important and crucial step of EDAs is how to build the probabilitic model P(x) to express the promising solutions. In EDAs for global continuous optimization problem, the Gaussian distribution is a common one. Some other complex models, like Gaussian mixture, histogram etc., are also used [8]. In order to construct a Gaussian pdf model of the promising solutions, we should obtain the statistical information of promising solutions. Hence, statistical techniques have been extensively applied to the optimization problems. Fortunately, these parameters can be efficiently computed by the maximum-likelihood estimations [6]. In the algorithm assume full independence, every variable is assumed independent of any variable. That is, the probability distribution $P(x_1, x_2, \ldots, x_D)$ of the vector $(x_1, x_2, \ldots, x_D)$ of m variables is assumed to consist of a product of the distributions of individual variables:

$$P^k(x_1, x_2, \cdots x_D) = \prod_{i=1}^{D} N(x_i \mid \mu_i^k, \sigma_i^k) \quad (1)$$

where μ_i^k is the mean and σ_i^k is the standard deviation of k-th generation and i-th variable. D is the dimension size. This is very suitable for calculation. Different from the discrete EDAs, the number of parameters to be estimated does not grow exponentially with D.

The pdf $N(x_i \mid \mu_i^k, \sigma_i^k)$ for variables xi is parameterized by the mean μ_i^k and the standard deviation σ_i^k, which is defined by

$$N(x_i^k \mid \mu_i^k, \sigma_i^k) = \frac{1}{\sigma_i^k \sqrt{2\pi}} e^{-\frac{(x_i - \mu_i^k)^2}{2\sigma_i^{k2}}} \tag{2}$$

Therefore, the probability distribution P(x1, x2,…, xD) of the vector (x1, x2,…, xD) of m variables is

$$P(x_1, x_2, \cdots x_D) = \prod_{i=1}^{m} \frac{1}{\sigma_i^k \sqrt{2\pi}} e^{-\frac{(x_i - \mu_i^k)^2}{2\sigma_i^{k2}}} \tag{3}$$

The parameters (μ_i^k, σ_i^k) can be estimated according to the selected best individuals. The parameters (μi, σi) can be updated every iteration.

The mean and standard deviation parameters of promising population can be computed adaptively by maximum likelihood technique according to the selected promising solutions.

$$\mu_i(k) = \frac{1}{NB} \sum_{n=1}^{NB} x_i^n(k) \tag{4}$$

$$\sigma_i^2(k) = \frac{1}{NB} \sum_{n=1}^{NB} (x_i^n(k) - \mu_i(k))(x_i^n(k) - \mu_i(k))^T \tag{5}$$

$\mu_i(k)$ is the mean of i-th variable in k-th iteration, NB is the selected individuals size. $\sigma_i^2(k)$ is the covariance of i-th variable in k-th iteration.

B. Probabilistic sampling

The probability sampling is used to generate new individuals using the learned probabilistic models instead of crossover or mutation operators. The sampling method depends on the type of probabilistic model and the characteristics of the problem. For normal pdf problem, a conversion can be used in order to convert the normal pdf to a standard normal pdf.

Supposing,

$$y = \frac{x - \mu}{\sigma} \tag{6}$$

The normal pdf about x is converted to a standard normal pdf about y.

$$N(x \mid \mu, \sigma) \to N(y \mid 0,\ 1) \tag{7}$$

The variable x can be calculated by

$$x = \sigma y + \mu \tag{8}$$

In the probability models, every variable (x1, x2,…, xm) is assumed independent of any variable. The mean and standard deviation of variable xi is μi and σi, when n→∞,

$$y = (\sum_{i=1}^{n} x_i - \sum_{i=1}^{n} \mu_i) / \sqrt{\sum_{i=1}^{n} \sigma_i^2} \to N(y,\ 0,1) \tag{9}$$

when n→∞, y→N(0, 1). We can select an appropriate n to generate a normal pdf for probability sampling.

C. Elitism strategy

Elitism strategy is an effective strategy to ensure the best individual(s) is selected as the next generation in EAs, because the best individual(s) maybe include the information of optimal solution[9]. Therefore, elitism can improves the convergence performance of EAs in many cases[10], and elitism has long been considered an effective method for improving the efficiency of EAs. This is achieved by simply copying the best individual(s) directly to the new generation[11]. However, the number of best individuals selected as the next generation must be handled properly and carefully otherwise may lead to premature convergence or can not improve the efficiency of algorithm.

$$Pop(k+1) = Elitism(k)_{NB} \cup Sample(k)_{NP-NB} \tag{10}$$

where *Elitism*() is the operator to copy the best solution to *Pop*(*k*+1), and *Sample*() is the sampling function. *N* is the population size, *NB* is the number of best individuals selected to build probability model.

D. Local search strategy

It is widely accepted that a local search procedure is efficient in improving the solutions generated by the EDA. Kinds of strategies are proposed to enhance the performance of EDA. In this paper, we use a chaos perturbation as the local search strategy[12]. The principle of perturbation is shown as figure 2.

The running of chaos operator is conditional. In this paper, the chaos perturbation is running under the condition of slower convergence.

$$\begin{cases} Pop_i^j(k) = Pop_i^j(k) + \eta z_i & \text{if < meet criteria >} \\ Pop_i^j(k) = Pop_i^j(k) & \text{else < does not meet criteria >} \end{cases} \tag{11}$$

where $Pop_i^j(k)$ is the i-th variable of j-th individual of k-th iteration. η is a perturbation coefficient. z_i is the chaotic variable, which can be generated by chaotic models. Many chaotic models can be used to generate chaotic variables [13], such as Logistic mapping, Cube mapping or infinite folding mapping. Logistic chaotic model is the most commonly used one, which folded within a limited number under a limited range. The logistic model is shown as follows.

$$z_{k+1} = \mu z_k (1 - z_k),\ n = 0,1,2,3... \tag{12}$$

It is a typical chaotic system. μ is the control variable, and a definite time series $z_1, z_2, z_3 \mathrm{L}$ can be generated by iteration for any $z_0 \in [0,1]$.

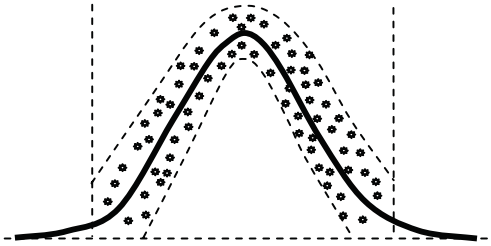

Fig. 2. The principle of perturbation

The model of infinite folding mapping is shown as equation (13):

$$z_{k+1} = \begin{cases} \sin\dfrac{2}{z_k} & k = 0,1,2\cdots \\ -1 < z_0 < 1 \end{cases} \tag{13}$$

The Cube mapping is shown as equation (14)

$$z_{k+1} = \begin{cases} 4{z_k}^3 - 3z_k & k = 0,1,2\cdots \\ -1 < z_0 < 1 \end{cases} \tag{14}$$

The Cube and infinite folding mapping do not need control variable u. The chaotic sequence distribution of logistic mapping is asymmetric. The probability density follows the distribution property of Chebyshev. The three mappings iterate 10000 times, and we have a statistics of the results. The distribution properties of the three mappings are shown in figure 3. The distribution property of infinite folding mapping is better other two mappings. Therefore, this paper adopts the infinite folding mapping from the common chaotic models to generate the chaotic variables. Moreover, the infinite folding mapping has little sensitivity to the initial value. Therefore, the infinite folding mapping is selected as chaotic variables generator.

The above way to generate new solutions does not take into account the feasibility of the solutions. The individuals could be out of the domain due to the perturbation. Therefore, a repair procedure is needed if illegal individuals are constructed.

$$\begin{cases} Pop_i^j(k) = Pop_i^j(k) \ \text{ if } Pop_i^j(k) \in [lb_i \ \ ub_i] \\ Pop_i^j(k) = ub_i \ \ or \ \ Pop_i^j(k) = lb_i \\ \qquad \text{else } Pop_i^j(k) > ub_i \ or \ Pop_i^j(k) < lb_i \end{cases} \tag{15}$$

$[lb_i \ \ ub_i]$ is the domain of *i*-th variable.

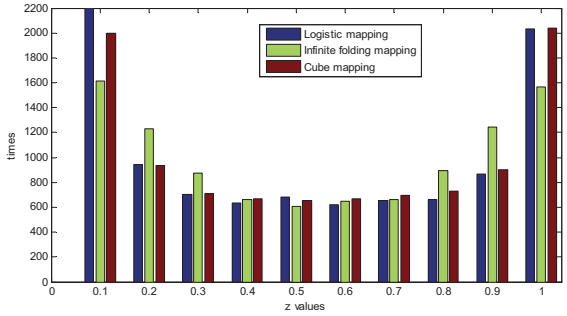

Fig. 3. Distribution property of Logistic, Cube and infinite folding mapping

E. Procedure of CEEDA

With the design above, the procedure of the ALREEDA is illustrated as following.

Begin

Initialization: Set parameters: Max_FEs , *NP*, *NB*, w_max , w_min , σ_i^0 , $[lb_i \ \ ub_i]$, and generate population *Pop*(0).

While(stop criteria ?)

Evaluation: Calculate the fitness of all individuals, and store the elitism.

Statistical information obtaining: Select NB individuals to estimate the parameter of the probabilistic model, and update the parameter.

$$\mu_i(k) = \frac{1}{N}\sum_{n=1}^{NB} x_i^n(k)$$

$$\sigma_i^2(k) = \frac{1}{N}\sum_{n=1}^{NB}(x_i^n(k) - \mu_i(k))(x_i^n(k) - \mu_i(k))^T$$

$$\sigma_i(k) = w\sigma_i(k) + (1-w)\sigma_i(k-1)$$

Probabilistic model building: According to estimated parameters, build the probabilistic model of each variable x_i

$$P(x_1, x_2, \cdots x_m) = \prod_{i=1}^{m} \frac{1}{\sigma_i^k\sqrt{2\pi}} e^{-\frac{(x_i-\mu_i^k)^2}{2\sigma_i^{k2}}}$$

Probabilistic sampling: Make use of the sampling technology sampling (*NP-NB*) individuals.

Elitism strategy: Combine the sampling individuals with elitism and generate new *Pop*(*k*).

$$Pop(k) = Elitism(NB)_k \cup Sample(N-NB)_k$$

Chaos perturbation (perturbation criteria ?):

$Pop_i^j(k) = Pop_i^j(k) + \eta z_i$

k=*k*+*1*

End While

End Begin

III. SIMULATION EXPERIMENTS

To testify the performance and scalability of the proposed algorithm, we use a special function f7 with complex structure of CEC08. The concrete formulas of functions are shown as follows, and we also give a graphical exposition to express the complexity of the functions.

$$f_7(\mathbf{x}) = \sum_{i=1}^{D} fractal1D(x_i + twist(x_{(i \bmod D)+1}))$$

$$twist(y) = 4(y^4 - 2y^3 + y^2)$$

$$Fractal1D(x) \approx$$

$$\sum_{k=1}^{3} \sum_{1}^{2^{k-1}} \sum_{1}^{ran2(o)} doubledip(x, ran1(o), \frac{1}{2^{k-1}(2 - ran1(o))})$$

$$doubledip(x,c,s) = \begin{cases} (-6144(x-c)^6 + 3088(x-c)^4 \\ \quad -392(x-c)^2 + 1)s, x \in (-0.5, 0.5) \\ 0, \text{ otherwise} \end{cases}$$

$\mathbf{x} = (x_1, x_2, \cdots, x_D)$, D is the dimensions, $x_i \in [-1,1]$. ran1(o) is a pseudorandomly chosen funciton, with seed o and equal probability from the interval [0,1], and having the precision of double. ran2(o) is also pseudorandomly chosen fucntion, with seed o and equal probability from the set {0,1,2}. fractal1D(x) is an approximation to a recursive algorithm; it does not take account of wrapping at the boundaries, or local re-seeding of the random generators. f_7 is a multimodal non-separable function. The global minimal is unknown.

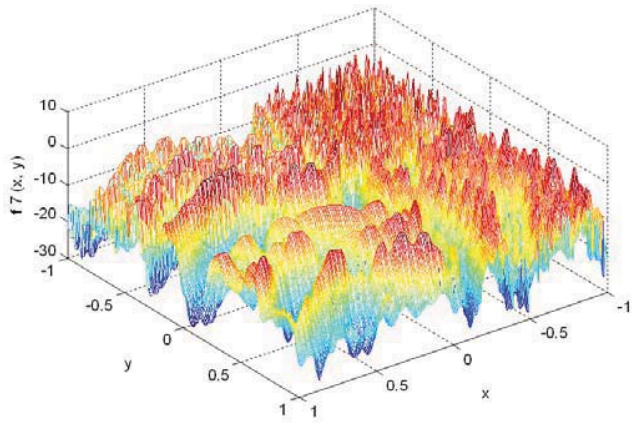

Fig. 4. The graph of f7

We can see from the figure 4, f7 is a very complex function. It looks like mountain range profile. There are many bigger maintains and also many narrow and dense peaks. The global optimum of f7 is unknown so far.

In the iteration optimization algorithm, such as EAs, the maximal iteration number (Max_FEs) is an essential parameter [14], while it is maybe varied. The Max_FEs is set to 3E+6 in this paper. For EDAs, the population size NP and the promising solution number NB are important except for Max_FEs. It is obvious that for an easy problem, a small value of NP is sufficient, but for difficult problems, a large value of NP is recommended in order to avoid trapping to a local optimum. The large NP may provide better optimization with larger calculation [15]. However, it is maybe vary from problem to problem. We pay attention to the performance of EDA instead of the population size. We provide some choices (100, 200, 300, 500) to obtain better performance for the CEEDA. We have a comparisons of different population size, and select the best population size as the final decision of the algorithm on the problem with the given problem size. The NB is also an important parameter for the probabilistic model learning of EDA, and we also have a comparison to determine a proper NB. In the comparisons, the error recorded finally is the absolute margin between the fitness of the best solution found and the fitness of the global optimum.

In figure 5, it is the testing result when the dimension D is 100 and has different NP and NB.

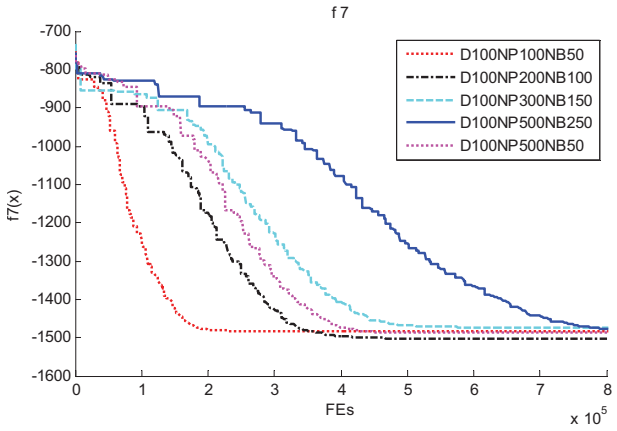

Fig. 5. Convergence graph for f_7 (D=100, different NP)

We also do some tests when the dimension D is 500 in figure 6. Firstly, we test the algorithm under different population size 100, 200 and 500. According to the performance of the algorithm, the population size is selected as 500. 50 is a suitable value for NB according to the performance of the algorithm on the benchmarks.

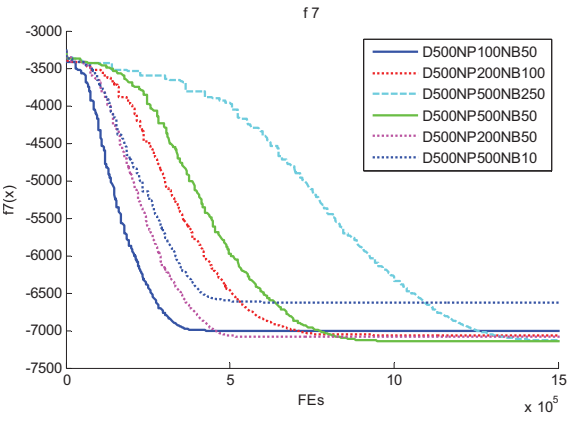

Fig. 6. Convergence graph for f_7 (D=500, different NP)

We also do a small test to testify the effect of the elitism strategy. The optimization process partial enlarge graph of f_7 are shown in figure 7. f_7 is the most obvious one. The optimization process is concussive without elitism strategy due to the complex f7 function, though the optimization is convergent finally. The convergence is smooth and steady when the elitism strategy is added to the algorithm.

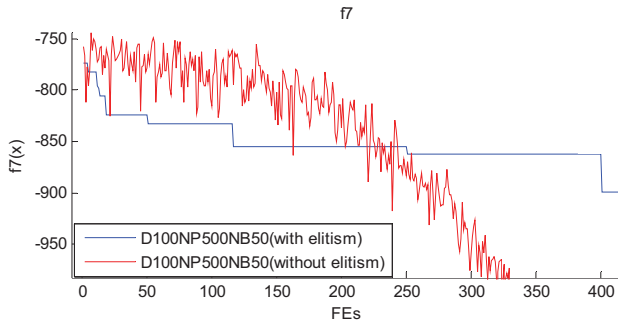

Fig. 7. The optimization process partial enlarge graph of f_7

We also compare CEEDA with other EDA. LSEDA-gl is a robust univariate EDA, which is proposed for large scale optimization and has good performance. In LSEDA-gl, an effective sampling under mixed Gaussian and Levy probability distribution is introduced to balance optimization and learning. And a restart mechanism is used to stop some variables shrinking dramatically to zero solely.

The convergent graph on 1000-D of the two algorithms is shown in figure 8. From the figure we can see the convergent process of CEEDA is gentle than LSEDA-gl. In comparison, CEEDA outperforms LSEDA-gl even though the convergence is kindly.

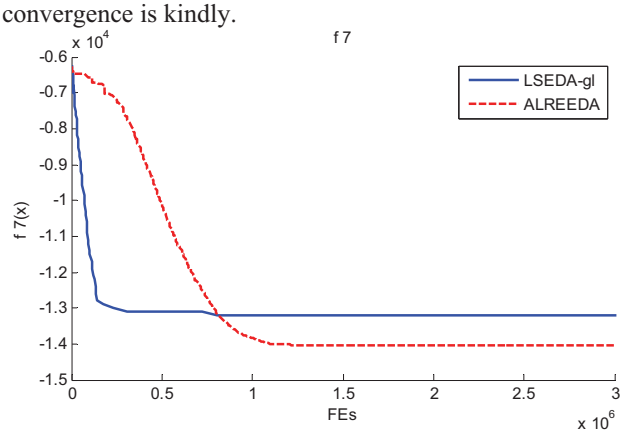

Fig. 8. Convergence graph for f_7 (D=1000)

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China (under Grant 61174044). The authors want to thank Ke Tang for providing the source code of the benchmarks and competition result of CEC'08 large scale optimization at website (http://nical.ustc.edu.cn/cec08ss.php).

REFERENCES

[1] H. Miuhlenbein and G. Paaß, "From Recombination of Genes to the Estimation of Distributions I. Binary Parameters", in *Proceedings of the 4th International Conference on Parallel Problem Solving from Nature*, pp. 178-187, London, UK,1996.

[2] S. Droste, "A rigorous analysis of the compact genetic algorithm for linear functions", *Natural Computing*, vol. 5, No. 3, pp. 257-283, 2006.

[3] M.E.L.D. Platel, S. Schliebs and N. Kasabov, "Quantum-inspired evolutionary algorithm: a multimodel EDA", *IEEE Transactions on Evolutionary Computation*, vol. 13, No. 6, pp. 1218--1232, 2009.

[4] J. Yang, H. Xu and P. Jia, "Effective search for Pittsburgh learning classifier systems via estimation of distribution algorithms", *Information Sciences*, vol. 198, pp. 100 - 117, 2012.

[5] X. Huang, P. Jia and B. Liu, "Controlling chaos by an improved estimation of distribution algorithm", *Mathematical and Computational Applications*, vol. 15, No. 5, pp. 866-871, 2010.

[6] P.A. Bosman and D. Thierens, *Numerical optimization with real-valued estimation-of-distribution algorithms, in Scalable Optimization via Probabilistic Modeling*. 2006, Springer: Berlin Heidelberg. p. 91--120.

[7] S. Muelas, A. Mendiburu, A. LaTorre and J. Peña, "Distributed Estimation of Distribution Algorithms for continuous optimization: How does the exchanged information influence their behavior?", *Information Sciences*, vol. 268, No. pp. 231-254, 2014.

[8] J. Sun, Q. Zhang and E.P.K. Tsang, "DE/EDA: A new evolutionary algorithm for global optimization", *Information Sciences*, vol. 169, No. 3‐4, pp. 249 - 262, 2005.

[9] E. Popovici and K. Jong, "The dynamics of the best individuals in co-evolution", *Natural Computing*, vol. 5, No. 3, pp. 229-255, 2006.

[10] C.W. Ahn and R.S. Ramakrishna, "Elitism-based compact genetic algorithms", *IEEE Transactions on Evolutionary Computation*, vol. 7, No. 4, pp. 367--385, 2003.

[11] I.J. Leno, S.S. Sankar, M.V. Raj and S.G. Ponnambalam, "An elitist strategy genetic algorithm for integrated layout design", *The International Journal of Advanced Manufacturing Technology*, vol. 66, No. 9-12, pp. 1573-1589, 2013.

[12] Y. Pei, "Chaotic Evolution: fusion of chaotic ergodicity and evolutionary iteration for optimization", *Natural Computing*, vol. 13, No. 1, pp. 79-96, 2014.

[13] M.S. Tavazoei and M. Haeri, "Comparison of different one-dimensional maps as chaotic search pattern in chaos optimization algorithms", *Applied Mathematics and Computation*, vol. 187, No. 2, pp. 1076-1085, 2007.

[14] W. Dong, T. Chen, P. Tino and X. Yao, "Scaling Up Estimation of Distribution Algorithms For Continuous Optimization", *IEEE Transactions on Evolutionary Computation*, vol. PP, No. 99, pp. 1-26, 2011.

[15] H. Karshenas, R. Santana, C. Bielza and P. Larra N Aga, "Regularized Continuous Estimation of Distribution Algorithms", *Applied Soft Computing*, vol. 13, No. 5, pp. 2412-2432, 2013