An Efficient Real-coded Genetic Algorithm for Real-Parameter Optimization

Zhi-Qiang Chen
Faculty of Engineering, University of Fukui
Fukui-shi, Japan

Rong-Long Wang
Faculty of Engineering, University of Fukui
Fukui-shi, Japan

***Abstract*—In this paper, we present an efficient real-coded genetic algorithm. In the proposed genetic algorithm model, crossover and mutation behaviors are performed by *similarity* between individuals. The proposed real coded genetic algorithm is compared with three existing genetic algorithms. A set of 18 test problems available in the global optimization literature is used to evaluate the performance of proposed genetic algorithm. The comparative study shows that the proposed genetic algorithm performs quite well and outperforms other algorithms.**

Keywords- Genetic Algorithm; Function Optimization; Real-Coded

I. INTRODUCTION

Genetic algorithm (GA) method, a robust and efficient search technique, has been used to many engineering applications since it was introduced. The popularity of this method is based on simply solving multidimensional and multimodal optimization problems without requiring any additional information such as the gradient of an objective function. Although the origin of this method proposed binary number for encoding, over the past ten years, there have been a surge of studies related to real-coded genetic algorithms (RCGA) for continue space problem [1~11]. In RCGAs, crossover has always been considered to be the fundamental search operator. A lot of efforts have been put into the development of sophisticated real coded crossover operators to improve the performances of RCGAs for real-parameter optimization. A serial of crossover operators have been presented such as Heuristic crossover [1], Flat crossover [2], Arithmetical crossover [3], Blend crossover(BLX-α) [4], Simulated binary crossover (SBX) [7], Unimodal normal distribution crossover operator (UNDX) [8], Simplex crossover operator(SPX) [9], Parent centric crossover operator(PCX) [10], and Laplace Crossover [11].

Besides the recombination operator, researchers have also realized the importance of the genetic algorithm model for real-parameter optimization. Many generation alternation models have been proposed, for instance, SGA [12], IGS [13], SS [14], CHC [15], ER [16], MGG [17], and G3 [10]. MGG is a commonly-used model originally for RCGAs [18] [19]. It is a steady-state model. G3 is a modification of MGG and only investigated on three commonly used test problems in [10].

In order to improve the global and local search ability of binary genetic algorithm, we proposed a genetic algorithm model called CGA [20], in which crossover and mutation behaviors are performed by *difference-degree* between individuals instead of given probability. The CGA model was applied to some combinatorial optimization problems [21, 22]. In this paper, based on the CGA model, we propose a new generation alternation model called *r*-CGA for the real-parameter optimization.

By combining *r*-CGA model with UNDX and *v*-NUM (a variant of Non-Uniform Mutation), we define a genetic algorithm called *r*-CGA+UNDX+*v*-NUM. The proposed genetic algorithm is compared with three existing genetic algorithms (MMG+UNDX, SGA+LX+NUM and MMG+ SPX). A set of 18 test problems available in the global optimization literature [11, 23] are used to evaluate the performance of the proposed genetic algorithm.

II. PROPOSED REAL–CODED GENETIC ALGORITHM

A. Generation Alternation Model

In [20], we proposed an efficient genetic algorithm called CGA. In CGA, only these parents whose *difference-degree* is larger than a given *threshold* produce offspring; other parents are performed mutation to keep the diversity. The *difference-degree* is an important parameter denoting the similarity between two chromosomes. The original CGA is proposed to solve the combinatorial optimization problems. In this section, based on the CGA, we propose a new generation alternation model called *r*-CGA for real coded genetic algorithm.

In the proposed *r*-CGA, we design a new computing method of the *difference-degree*. Given two chromosomes $\vec{x}_1$ and $\vec{x}_2$, the *difference-degree* between $\vec{x}_1$ and $\vec{x}_2$ is defined as:

$$d_i = \frac{\|\vec{e}_1 - \vec{e}_2\|}{2} \tag{1}$$

where $\|\vec{e}_1 - \vec{e}_2\|$ is the distance between the vectors $\vec{e}_1$ and $\vec{e}_2$, $\vec{e}_1$ and $\vec{e}_2$ are computed as follows:

$$\begin{aligned} \vec{e}_1 &= \vec{x}_1 / \|\vec{x}_1\| \\ \vec{e}_2 &= \vec{x}_2 / \|\vec{x}_2\| \end{aligned} \tag{2}$$

Based on the triangle theorem,

$$0 \le \|\vec{e}_1 - \vec{e}_2\| \le \|\vec{e}_1\| + \|\vec{e}_2\| = 2 \tag{3}$$

It is clear that $d_i \in [0,1]$.

Another important parameter in the *r*-CGA is the given threshold D_s. D_s is decreased in every generation and defined as:

$$D_s(t+1) = \mu D_s(t) \quad (4)$$

where *t* expresses #*t* generation, $\mu \in (0,1)$ is a constant.

Let N_P be the population size, the *r*-CGA model is intertwined in the following manner:

Step1. Set the cooling ratio μ, and the initial value of the *threshold* D_s;

Step2. Create mating cool P_M with N_P individuals using tournament selection for crossover and mutation behaviors.

Step3. Produce N_P offspring according to the following sub-procedure:

① Set $N_C = 0$, N_C is a counter of offspring.

②Select $(N_P\text{-}N_C)/2$ uniformly at random pairs of individual as parents from the mating cool P_M.

③ Calculate *difference-degree* d_i of each parent pair using (1).

④ Apply crossover operator on every pair of parent whose *difference-degree* d_i is larger than *threshold* D_s. The pair of parent performed crossover generates two offspring. $N_C = N_C + 1$ after creating an offspring.

⑤ If $N_C = N_P$, all offspring replace last generation as a new generation and terminate this sub-procedure. Note that elite-preserving is applied when performing generation alteration.

⑥ If $N_C < N_P$, a mutation operator is applied on the parent pairs whose *difference-degree* d_i is smaller than *threshold* D_s. Note that the mutation operator is used to add the diversity of the mating pool and does not produce offspring.

⑦ Go to sub step ②.

Step4. Decrease *threshold* D_s by using (4).

Step5. Terminate this procedure if termination criterion is reached.

Step6. Go to step2.

B. Crossover and Mutation Operators

1) Crossover Operator

UNDX [8] is a well used crossover method, in which multiple parents are used to create two or more offspring solutions around the center of mass of these parents. A small probability is assigned to solutions away from the center of mass. In this paper, we use UNDX as crossover operator in our genetic algorithm. The UNDX crossover is as follows:

$$\vec{c}_1 = \vec{m} + z_1\vec{e}_1 + \sum_{k=2}^{n} z_k\vec{e}_k \quad (6)$$

$$c_2 = \vec{m} - z_1\vec{e}_2 - \sum_{k=2}^{n} z_k\vec{e}_k \quad (7)$$

$$\vec{m} = (\vec{p}_1 + \vec{p}_2)/2,$$
$$z_1 \sim N(0,\sigma_1^2), z_k \sim N(0,\sigma_2^2)(k = 2,...,n),$$
$$\sigma_1 = \alpha d_1, \sigma_2 = \beta d_2/\sqrt{n},$$
$$\vec{e}_1 = (\vec{p}_2 - \vec{p}_1)/|\vec{p}_2 - \vec{p}_1|, \vec{e}_i \perp \vec{e}_j (i, j = 1,...,n, i \neq j)$$

Here, *n* is the dimension of the variable. $\vec{p}_1$ and $\vec{p}_2$ are a pair of parent whose *difference-degree* is larger than *threshold* D_S. d_2 is the distance from $\vec{p}_3$ (a parent selected uniformly at random from the mating pool) to $\vec{p}_1 - \vec{p}_2$ and $d_1 = |\vec{p}_1 - \vec{p}_2|$. α and β is parameter defined by users. The commended settings are $\alpha = 0.5$, and $\beta = 0.35$, respectively.

2) Mutation Operator

Michalewicz's Non-Uniform Mutation (NUM) is one of the widely used mutation operators in real-coded genetic algorithm [3]. Here we proposed a modified non-uniform mutation operator (*v*-NUM) for real-coded genetic algorithm. For a point $\vec{x} = (x_1, x_2, ..., x_n)$, the mutated point $\vec{x}' = (x_1', x_2', ..., x_n')$ is created as follows:

$$x_i' = x_i + y(1-u)^b \quad (8)$$

$$y = \begin{cases} x_i^u - x_i & if \ r \leq 0.5, \\ x_i^l - x_i & otherwise \end{cases} \quad (9)$$

where u and r are both uniformly distributed random number in the interval [0, 1]; x_i^l and x_i^u are lower and upper bounds of the # i decision variable, respectively; b is a parameter determining the strength of the mutation operator and is set to 4 in this paper.

By combining *r*-CGA model with UNDX and *v*-NUM, we define a genetic algorithm called *r*-CGA+UNDX+*v*-NUM.

III. Simulations

A. Test Bed

We used a test-bed of 18 traditional benchmark functions (f_1~f_{18}) available in the global optimization literature [11, 23] to evaluate the performance of the proposed scheme. They are reported as following:

1. Sphere function (f_1)

$$\min_x f(x) = \sum_{i=1}^{n} x_i^2,$$
$$-5.12 \leq x_i \leq 5.12,\ x^* = (0,0,...,0),\ f(x^*) = 0.$$

2. Ellipsoid function (f_2)

$$\min_x f(x) = \sum_{i=1}^{n} (1000^{i-1/n-1} x_i)^2,$$

$$-5.12 \le x_i \le 5.12,\ x^* = (0,0,...,0),\ f(x^*) = 0.$$

3. *k*-tablet function (f_3)

$$\min_x f(x) = \sum_{i=1}^{k} x_i^2 + \sum_{i=k+1}^{n} (100 x_i)^2,$$

$$-5.12 \le x_i \le 5.12,\ x^* = (0,0,...,0),\ f(x^*) = 0.$$

4. Schewefel problem 3 (f_4)

$$\min_x f(x) = \sum_{i=1}^{n} | x_i | + \prod_{i=1}^{n} | x_i |,$$

$$-10 \le x_i \le 10, x^* = (0,0,...,0)\ and\ f(x^*) = 0.$$

5. Schewefel problem 4 (f_5)

$$\min_x f(x) = \max_x \{ | x_i |, 1 \le i \le n \},$$

$$-100 \le x_i \le 100, x^* = (0,0,...,0)\ and\ f(x^*) = 0.$$

6. Axis parallel hyper ellipsoid (f_6)

$$\min_x f(x) = \sum_{i=1}^{n} i x_i^2,$$

$$-5.12 \le x_i \le 5.12, x^* = (0,0,...,0)\ and\ f(x^*) = 0.$$

7. Zakharov's function (f_7)

$$\min_x f(x) = \sum_{i=1}^{n} x_i^2 + (\sum_{i=1}^{n} \frac{i}{2} x_i)^2 + (\sum_{i=1}^{n} \frac{i}{2} x_i)^4,$$

$$-5.12 \le x_i \le 5.12,\ x^* = (0,0,...,0)\ and\ f(x^*) = 0.$$

8. Exponential problem (f_8)

$$\min_x f(x) = -\exp(0.5 \sum_{i=1}^{n} x_i^2),$$

$$-1 \le x_i \le 1,\ x^* = (0,0,...,0)\ and\ f(x^*) = -1.$$

9. Ellipsoidal function (f_9)

$$\min_x f(x) = \sum_{i=1}^{n} (x_i - i)^2,$$

$$-n \le x_i \le n,\ x^* = (1,2,...,n)\ and\ f(x^*) = 0.$$

10. Ackley's problem (f_{10})

$$\min_x f(x) = -20 \exp(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2}\ )$$

$$-\exp(\frac{1}{n} \sum_{i=1}^{n} \cos(2\pi x_i)) + 20 + e,$$

$$-30 \le x_i \le 30,\ x^* = (0,0,...,0)\ and\ f(x^*) = 0.$$

12. Cosine mixture problem (f_{11})

$$\min_x f(x) = \sum_{i=1}^{n} x_i^2 - 0.1 \sum_{i=1}^{n} \cos(5\pi x_i),$$

$$-1 \le x_i \le 1,\ x^* = (0,0,...,0)\ and\ f(x^*) = -0.1n.$$

12. Griewank problem (f_{12})

$$\min_x f(x) = 1 + \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}}),$$

$$-600 \le x_i \le 600,\ x^* = (0,0,...,0)\ and\ f(x^*) = 0.$$

13. Levy and Montalvo problem 1 (f_{13})

$$\min_x f(x) = \frac{\pi}{n} (10 \sin^2(\pi y_1) +$$

$$\sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10 \sin^2(\pi y_{i+1})] (y_n - 1)^2),$$

$$where\ y_i = 1 + \frac{1}{4}(x_i + 1),$$

$$-10 \le x_i \le 10,\ x^* = (0,0,...,0)\ and\ f(x^*) = 0.$$

14. Levy and Montalvo problem 2 (f_{14})

$$\min_x f(x) = 0.1 (\sin^2(3\pi x_1) +$$

$$\sum_{i=1}^{n-1} (x_i - 1)^2 [1 + \sin^2(3\pi x_{i+1})] + (x_n - 1)^2 [1 + \sin^2(2\pi x_n)]),$$

$$-5 \le x_i \le 5,\ x^* = (0,0,...,0)\ and\ f(x^*) = 0.$$

15. Generalized penalized function 1 (f_{15})

$$\min_x f(x) = \frac{\pi}{n} (10 \sin^2(\pi y_1) +$$

$$\sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10 \sin^2(\pi y_{i+1})] + (y_n - 1)^2) + \sum_{i=1}^{n} u(x_i, 10, 100, 4),$$

$$where\ y_i = 1 + \tfrac{1}{4}(x_i + 1),$$

$$-10 \le x_i \le 10,\ x^* = (0,0,...,0)\ and\ f(x^*) = 0.$$

16. Generalized penalized function 2 (f_{16})

$$\min_x f(x) = 0.1 (\sin^2(3\pi x_1) +$$

$$\sum_{i=1}^{n-1} (x_i - 1)^2 [1 + \sin^2(3\pi x_{i+1})] +$$

$$(x_n - 1)^2) [1 + \sin^2(2\pi x_n)] + \sum_{i=1}^{n} u(x_i, 10, 100, 4),$$

$$-5 \le x_i \le 5,\ x^* = (0,0,...,0)\ and\ f(x^*) = 0.$$

In the problem 15 and 16, the penalty function u is given by the following expression:

$$u(x,a,k,m) = \begin{cases} k * pow((x-a), m) & if\ x > a, \\ -k * pow((x-a), m) & if\ x < -a, \\ 0 & oyherwise. \end{cases}$$

17. Bohachevsky function (f_{17})

$$\min_x f(x) = \sum_{i=1}^{n-1} (x_i^2 + 2x_{i+1}^2 - 0.3 \cos(3\pi x_i) - 0.4 \cos(4\pi x_{i+1}) + 0.7),$$

$$-5.12 \le x_i \le 5.12,\ x^* = (0,0,...,0),\ f(x^*) = 0.$$

18. Rastrigin function (f_{18})

TABLE I COMPARISIONS WITH OTHER ALGORITHMS

Pro	*r*-CGA+UNDX+ *v*-NUM			MMG+UNDX				MMG+SPX				SGA+LX+NUM	
	μ	N_P	*Least FEs* *Mean FEs* *Most FEs*	N_P	N_C	*Least FEs* *Mean FEs* *Most FEs*	R_{FEs}	N_P	N_C	*Least FEs* *Mean FEs* *Most FEs*	R_{FEs}	*Least FEs* *Mean FEs* *Most FEs*	R_{FEs}
f_1	0.999	30	2.96e+4 3.56e+4 4.02e+4	50	40	7.45e+4 8.03e+4 8.63e+4	2.26	10D	2D	4.05e+5 6.81e+5 8.00e+5	19.1	2.02e+5 2.22e+5 2.38e+5	6.24
f_2	0.9999	600	5.84e+5 6.32e+5 7.26e+5			-- -- -- --	--	30D	2D	1.84e+6 1.94e+6 2.06e+6	3.07	3.14e+5 3.39e+5 3.58e+5	0.54
f_3	0.9999	600	5.77e+5 9.54e+5 1.32e+6			-- -- -- --	--	30D	2D	1.83e+6 1.95e+6 2.16e+6	2.04	3.14e+5 3.41e+5 3.66e+5	0.36
f_4	0.9999	100	2.01e+5 2.53e+5 4.49e+5	300	100	2.65e+6 4.07e+6 4.07e+6	16.1	20D	2D	1.62e+6 1.65e+6 1.70e+6	6.52	4.33e+5 4.59e+5 4.84e+5	1.81
f_5	0.9999	100	2.41e+5 2.84e+5 3.60e+5	50	40	9.51e+5 1.32e+6 1.86e+6	4.64	20D	2D	1.74e+6 1.79e+6 1.89e+6	6.30	------	--
f_6	0.999	30	2.39e+5 3.22e+5 3.90e+5	50	40	2.10e+5 3.09e+5 3.97e+5	0.96	20D	2D	8.78e+5 9.08e+5 9.44e+5	2.94	2.36e+5 2.53e+5 2.73e+5	0.79
f_7	0.999	50	1.04e+5 1.16e+5 1.33e+5	50	40	2.40e+5 2.56e+5 2.84e+5	2.22	20D	2D	8.80e+5 9.11e+5 9.47e+5	7.85	------	--
f_8	0.999	50	2.14e+4 2.44e+4 2.97e+4	50	40	5.80e+4 6.11e+4 6.52e+4	2.50	20D	2D	6.02e+5 6.32e+5 6.56e+5	25.9	1.58e+5 1.70e+5 1.84e+5	6.97
f_9	0.99	30	4.78e+4 6.30e+4 9.44e+4	50	40	1.06e+5 1.12e+5 1.19e+5	1.78			-- -- --	--	2.54e+5 2.71e+5 2.96e+5	4.3
f_{10}	0.999	50	1.02e+5 1.08e+5 1.16e+5	50	120	3.47e+5 3.63e+5 3.88e+5	3.36	20D	2D	1.51e+6 1.54e+6 1.56e+6	14.3	7.33e+5 8.14e+5 9.44e+5	7.54
f_{11}	0.999	100	3.12e+4 3.74e+4 4.17e+4	50	100	1.46e+5 1.58e+5 1.77e+5	4.22	20D	2D	7.66e+5 7.83e+5 8.07e+5	20.9	1.98e+5 2.16e+5 2.37e+5	5.78
f_{12}	0.9999	100	9.56e+4 1.56e+5 3.06e+5	200	100	9.43e+5 1.04e+6 1.11e+6	6.67	20D	2D	1.04e+6 1.07e+6 1.11e+6	6.86	2.62e+5 3.94e+5 5.00e+5	2.53
f_{13}	0.99	30	5.07e+3 8.54e+3 1.24e+4	50	40	1.61e+5 4.45e+5 8.49e+5	52.1	20D	2D	2.68e+5 4.39e+5 6.46e+5	51.4	2.54e+3 8.67e+3 2.19e+4	1.02
f_{14}	0.99	50	4.65e+4 4.85e+4 5.19e+4	100	100	3.30e+5 3.58e+5 3.80e+5	7.38	20D	2D	7.22e+5 7.42e+5 7.69e+5	15.3	1.70e+5 1.80e+5 1.91e+5	3.71
f_{15}	0.99	50	3.94e+4 4.15e+4 4.66e+4	50	40	8.96e+4 9.84e+4 1.12e+5	2.37	20D	2D	6.68e+5 6.89e+5 7.15e+5	16.6	1.48e+5 1.75e+5 1.95e+5	4.22
f_{16}	0.99	50	4.60e+4 4.88e+4 5.24e+4	50	100	1.83e+5 1.95e+5 2.16e+5	4.0	20D	2D	7.21e+5 7.41e+5 7.81e+5	15.2	1.76e+5 1.93e+5 2.06e+5	3.95
f_{17}	0.9999	100	5.95e+4 7.42e+4 9.16e+4	300	200	1.88e+6 2.41e+6 2.98e+6	32.5	20D	2D	9.55e+5 9.74e+5 9.97e+5	13.1	2.47e+5 2.73e+5 2.94e+5	3.68
f_{18}	0.9999	100	6.89e+4 9.06e+4 1.49e+5	300	200	3.00e+6 4.23e+6 9.00e+6	46.7	50D	2D	3.48e+6 3.56e+6 3.66e+6	39.3	3.30e+5 3.80e+5 4.13e+5	4.2

$$\min_x f(x) = 10n + \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i)) ,$$
$$-5.12 \le x_i \le 5.12,\ \ x^* = (0,0,...,0),\ f(x^*) = 0.$$

B. *Experiment*

25 independent runs are carried out for each problem. The dimension of the variable in the entire problems is fixed to 30. Initial individual is initialized uniformly at random within the search space. All experiments are performed to evaluate the performance of the proposed genetic algorithm at a PC station. The evaluation criterion is the number of the function evaluations (*FEs*) to achieve the fixed accuracy (within the error value 10^{-7}). To evaluate the performance of the proposed algorithm, we compare our *r*-CGA+UNDX+*v*-NUM method with three existing real-coded genetic algorithms: MMG+ UNDX [10], MMG+SPX [9], SGA +LX+NUM [11].

In case of real coded GA, parameter tuning is generally more difficult as compared to binary coded GA due to the simple reason that the numbers of tunable parameters occurring in a real coded GA are usually more than that occurring in binary GA. This results in more possible combinations of parameter settings in real coded GAs than binary GAs. It becomes very challenging to suggest common fixed values of various parameters for the entire suit. To achieve this goal, we have carried out extensive experiments for four algorithms. The initial value of the *threshold* D_s and tournament size for *r*-CGA+UNDX+*v*-NUM are fixed to 0.5 and 3, respectively. The cooling ratio μ and population size N_P are set by Table I. MMG+UNDX and MMG+SPX are two steady state models that uses UNDX and SPX crossover, respectively. The population size (N_P) and the number (N_C) of offspring generated from a group of parents are set by Table I. SGA +LX+NUM is a simple genetic algorithm model with tournament select that uses LX crossover and its parameters are set by [11].

All 25 runs for every test problem achieve within the error value 10^{-7}, except that the success rate is 84% for f_{12} using *r*-CGA+UNDX+*v*-NUM, 88% for f_{18} using MMG+UNDX. The *least, mean* and *most FEs* in 25 runs for every problem using different algorithm are recorded in the Table I, respectively. R_{FEs} in the Table I is defined as: (*Mean FEs* using other algorithms)/(*Mean FEs* using *r*-CGA+UNDX+ *v*-NUM). Thus, $R_{FEs} > 1$ expresses that the proposed algorithm is superior to its competitor. From the table, we can know that the proposed algorithm performs quite well and outperforms other algorithms in almost all problems.

IV. CONCLUSIONS

In this study we defined an efficient real coded genetic algorithm called *r*-CGA+UNDX+*v*-NUM. The experiments are performed on a set of 18 benchmark problems available in global optimization literature. Three different algorithms (SGA+LX+NUM, MMG+SPX, MMG+UNDX) are employed to compare the performance of the proposed algorithm. Simulation results show that the proposed *r*-CGA+UNDX+*v*-NUM outperforms other algorithms.

REFERENCES

[1] A.H. Wright, "Genetic algorithms for real parameter optimization", in: G.J.E. Rawlins (Ed.), Foundations of Genetic Algorithms I, Morgan Kaufmann, San Mateo, 1991, pp. 205−218.

[2] N.J. Radcliffe, "Equivalence class analysis of genetic algorithms", Complex Systems 2 (5) , 1991, pp.183-205.

[3] Z. Michalewicz, "Genetic Algorithms + Data Structures = Evolution Programs", Springer-Verlag, New York, 1992.

[4] L.J. Eshelman, J.D. Schaffer, "Real-coded genetic algorithms and interval schemata", in: D.L. Whitley (Ed.), Foundation of Genetic Algorithms II, Morgan Kaufmann, San Mateo, CA, 1993, pp. 187-202.

[5] H. Muhlebein, D. Schlierkamp-Voosen, "Predictive models for breeder genetic algorithms in continuous parameter optimization", Evolutionary Computation 1 (1) , 1993, pp. 25-49.

[6] H.M. Voigt, H. Muhlenbein, D. Cvetkovic, "Fuzzy recombination for the breeder genetic algorithms", in: L.J. Eshelmen (Ed.), Proceedings of the 6th International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1995, pp. 104-111.

[7] K. Deb, R.B. Agrawal, "Simulated binary crossover for continuous search space", Complex Systems 9 , 1995, pp. 115-148.

[8] I. Ono, S. Kobayashi, "A real-coded genetic algorithm for function optimization using unimodal normal distribution crossover", in: T. Back (Ed.), Proceedings of the Seventh International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA,1997, pp. 246−253.

[9] S. Tsutsui, M. Yamamura, T. Higuchi, "Multi-parent recombination with simplex crossover in real-coded genetic algorithms", in: W. Banzhaf, J. Daida, A. Eiben, M. Garzon, V. Honavar, M. Jakiela, R. Smith (Eds.), Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-1 1999), 1999, pp. 657−664.

[10] K. Deb, A. Anand, D. Joshi, "A computationally efficient evolutionary algorithm for real-parameter evolution", Evolutionary Computation Journal 10 (4) , 2002, pp. 371-395.

[11] K. Deep, M. Thakur, "A new crossover operator for real coded genetic algorithms", Applied Mathematics and Computation 188 , 2007, 895-911.

[12] D. E. Goldberg. "Genetic Algorithms in Search, Optimization, and Machine Learning", AddisonWsley Publishing Company Inc., 1989.

[13] D. H. Ackley. "An empirical study of bit vector function optimization", *Genetic Algorithms and Simulated Annealing*, vol 1, 1987, pp.170–204.

[14] G. Syswerda. "A study of reproduction in generational and steady-state genetic algorithms", *Foundations of Genetic Algorithms*, 1991, pp. 94 – 101.

[15] L. J. Eshelman. "The chc adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination", *Foundations of Genetic Algorithms*, 1991, pp. 265 –283.

[16] D. Thirens and D. E. Goldberg. "Elitist recombination: An integrated selection recombination ga", In *Proceedings of the IEEE Congress on Evolutionary Computation*, 1994, pp. 508 – 512.

[17] H. Satoh, M. Yamamura, and S. Kobayashi, "Minimal generation gap model for Gas considering both exploration and exploitation". *Proceedings of the IIZUKA: Methodologies for the Conception, Design, and Application of Intelligent Systems*. 1996, pp.494-497.

[18] I. Ono, and S. Kobayashi, "A Real-coded Algorithms for Function Optimization Using Unimodal Normal Distribution Crossover", *Proc. 7th Int.Conf. on Genetic Algorithms*, 1997, pp.246-253.

[19] I. Ono, and S. Kobayashi, and K. Yoshida, "Optimal Lens Design by Real-coded Genetic Algorithms using UNDX", *Computer Methods in Applied Mechanics and Engineering*, No.186, 2000, pp. 1146-1155.

[20] R. L. Wang, "A genetic algorithm for subset sum problem", Neurocomputing, Vol.57, 2004, pp.463-468.

[21] R. L. WANG and Kozo OKAZAKI, "Solving Graph Planarization Problem Using an Improved Genetic Algorithm", IEICE TRANS, FUNDAMENTALS, VOL.E89-A, NO.5 May 2006.

[22] R. L. WANG and Kozo OKAZAKI, "An improved genetic algorithm with conditional genetic operators and its application to set-covering problem", Soft Comput vol 11, 2007, pp. 687-694.