A Robust Real-coded Genetic Algorithm using an Ensemble of Crossover Operators

Jeonghwan Gwak
School of Information and Communications,
Gwangju Institute of Science and Technology (GIST),
Gwangju 500-712, Korea
gawk@gist.ac.kr

Moongu Jeon*
School of Information and Communications,
Gwangju Institute of Science and Technology (GIST),
Gwangju 500-712, Korea
mgjeon@gist.ac.kr

ABSTRACT
Although a lot of crossover operators have been developed for genetic algorithms (GAs), there is not much research on combining different crossover operators to form robust real-coded GAs. In this work, we propose an ensemble of crossover operators which is realized by two different parallel populations. The effectiveness of the proposed method is evaluated for traditional 6 benchmark functions. Results demonstrated that the proposed method has good generalization performance.

Categories and Subject Descriptors
I.2.8 [**Problem Solving, Control Methods, and Search**]: Heuristic Methods

General Terms
Algorithms, Design, Reliability

Keywords
Real-coded genetic algorithms, Ensemble learning, Blend crossover, Simulated binary crossover

1. INTRODUCTION
As the no free lunch theorem [1] suggests, there is no such algorithm that outperforms all others on a huge collection of optimization problems. In genetic algorithms (GAs), the crossover operator producing two descendants by combining the traits of two parents is regarded as the primary search operator. It can be found in [2] that in real-coded GAs (RCGAs), different crossover operators have different performance according to their exploration and exploitation features. Therefore, it is unlikely to devise RCGA only using a single crossover operator to perform well on various optimization problems. Hence, it is natural to consider an ensemble of different crossover operators and parameters to obtain benefits from the strength of each crossover operator.

2. PROPOSED RCGA BASED ON THE ENSEMBLE OF CROSSOVER OPERATORS
In this work, two widely used real-coded crossover operators, the blend crossover (BLX) [3], especially the version BLX-0.5, and simulated binary crossover (SBX) [4], were adopted for implementation of the ensemble idea. Most GAs generally apply crossover operators for all variables when the generated random number is less than or equal to the given crossover probability (p_c).

* Corresponding Author

GECCO'13 Companion, July 6–10, 2013, Amsterdam, The Netherlands.
ACM 978-1-4503-1964-5/13/07.

However, from several experimental simulations, we found that it is also important to decide whether each variable undergoes crossover for the practical application of GAs to solve multidimensional problems (see Observation 1 in Section 3). In this regard, we classify each BLX and SBX into two types: For each variable of *D* dimensional search space, 1) $BLX^{1.0}$: All genes (i.e., variables) are crossed over within the gene-wise BLX operation, 2) $BLX^{0.5}$: Only 50% genes (on average) are crossed over within the gene-wise BLX operation, 3) $SBX^{1.0}$: All genes are crossed over within the gene-wise SBX operation, 4) $SBX^{0.5}$: Only 50% genes (on average) are crossed over within the gene-wise SBX operation.
The flowchart of RCGA based on the proposed ensembles of crossover operators is depicted in Fig 1.

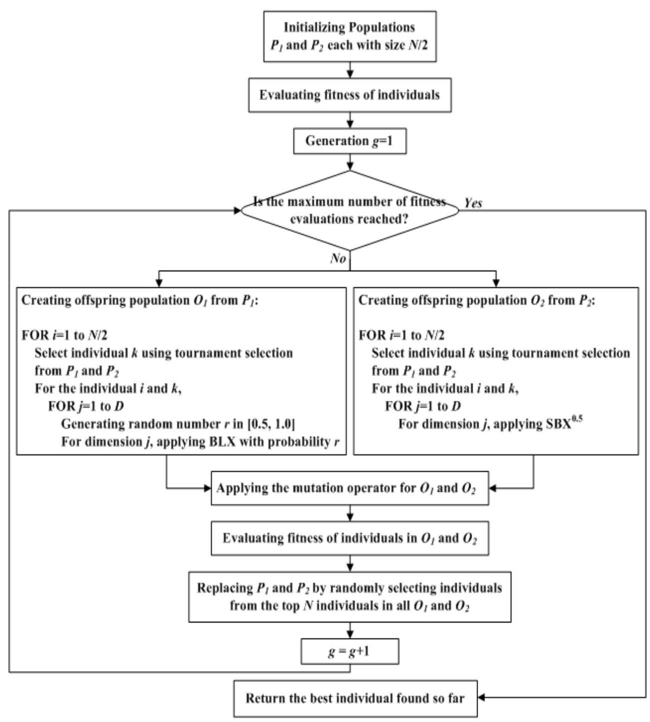

Figure 1. Flowchart of RCGA based on the Proposed Method

Starting from the initial populations P_1 and P_2 with each size of *N/2*, each individual in P_1 and P_2 is mated with an individual selected using tournament selection with the tournament size of 2. Next, the ensemble of crossover operators with $p_c = 1.0$ is applied. Then, the non-uniform mutation [5] with the probability of 0.5/*D* (where *D* is the number of decision variables) is performed. As a result, a set of $2 \times N$ offspring is generated and its top *N* individuals are chosen and stored to construct the offspring population.

3. EXPERIMENTAL RESULTS

To evaluate the performance of the proposed algorithm, we compare it with the four GAs using different crossover operators $BLX^{0.5}$, $BLX^{1.0}$, $SBX^{0.5}$ and $SBX^{1.0}$ for the following 6 traditional benchmark functions with $D = 50$ in Table 1. For the functions, the maximum number of fitness evaluations was set to $10,000 \times D$.

Table 1. The 5 Benchmark Functions

Function	Name	Range	Fitness Optimum
F1	Ellipsoidal Function	$[-100, 100]^D$	0
F2	Schwefel's Problem 1.2	$[-100, 100]^D$	0
F3	Rosenbrock's Function	$[-30, 30]^D$	0
F4	Rastrigin's Function	$[-5.12, 5.12]^D$	0
F5	Ackley's Function	$[-32, 32]^D$	0
F6	Griewank's Function	$[-600, 600]^D$	0

The results for 100 independent runs are summarized in Table 2, where "Avg." indicates the average of best fitness values for the 100 runs, and "Std. Dev." stands for their standard deviation, and finally "Rank" is the performance order of the algorithms. For each function, the best algorithm among $BLX^{0.5}$, $BLX^{1.0}$, $SBX^{0.5}$, and $SBX^{1.0}$ are highlighted in boldface, and the algorithm with the shaded means it has the best performance in terms of fitness accuracy compared to the optimum.

Table 2. Experimental Results

	GA Type	Avg.	Std. Dev.	Rank
F1	GA with $BLX^{0.5}$	3.03E-14	8.95E-14	4
	GA with $BLX^{1.0}$	**1.37E-25**	**3.36E-25**	**2**
	GA with $SBX^{0.5}$	1.04E-16	5.09E-16	3
	GA with $SBX^{1.0}$	3.91E-09	1.75E-09	5
	Proposed Method	8.66E-32	6.12E-31	1
F2	GA with $BLX^{0.5}$	9.57E-13	3.35E-12	4
	GA with $BLX^{1.0}$	**1.78E-27**	**3.70E-27**	**2**
	GA with $SBX^{0.5}$	3.50E-15	3.10E-14	3
	GA with $SBX^{1.0}$	4.13E-08	1.68E-08	5
	Proposed Method	5.80E-30	5.80E-29	1
F3	GA with $BLX^{0.5}$	7.65E+01	4.13E+01	4
	GA with $BLX^{1.0}$	**5.21E+01**	**2.21E+01**	**1**
	GA with $SBX^{0.5}$	7.43E+01	3.31E+01	3
	GA with $SBX^{1.0}$	1.41E+02	1.50E+02	5
	Proposed Method	6.58E+01	4.97E+01	2
F4	**GA with $BLX^{0.5}$**	**1.33E-04**	**1.33E-03**	**2**
	GA with $BLX^{1.0}$	1.53E+01	3.14E+00	4
	GA with $SBX^{0.5}$	9.95E-03	9.95E-02	3
	GA with $SBX^{1.0}$	2.09E+01	3.36E+00	5
	Proposed Method	1.13E-13	1.13E-12	1
F5	GA with $BLX^{0.5}$	3.44E-09	4.90E-09	2
	GA with $BLX^{1.0}$	2.00E+01	0.00E+00	4
	GA with $SBX^{0.5}$	**2.79E-09**	**2.33E-09**	**1**
	GA with $SBX^{1.0}$	2.00E+01	0.00E+00	4
	Proposed Method	2.58E-06	1.38E-05	3
F6	**GA with $BLX^{0.5}$**	**1.10E-12**	**3.81E-13**	**1**
	GA with $BLX^{1.0}$	1.60E-08	3.77E-09	3
	GA with $SBX^{0.5}$	3.45E-04	1.72E-03	4
	GA with $SBX^{1.0}$	5.67E-02	1.89E-02	5
	Proposed Method	4.47E-11	1.09E-11	2

Observation 1. By comparing the performance of $BLX^{0.5}$, $BLX^{1.0}$, $SBX^{0.5}$ and $SBX^{1.0}$, it can be found that (i) $BLX^{0.5}$ was best for F4 and F6, (ii) $BLX^{1.0}$ was the best for F1, F2 and F3, and (iii) $SBX^{0.5}$ was the best for F5, (iv) $SBX^{1.0}$ always was the worst for all functions. This observation led us to devise the ensemble of crossover operators (i.e., $BLX^{0.5}$, $BLX^{1.0}$, and $SBX^{1.0}$).

Observation 2. When it comes to the performance of the proposed method, it showed (i) the first ranked (i.e., the best performance) for F1, F2 and F4, (ii) the second ranked for F3 and F6, and (iii) the third ranked for F5. From this observation, we can conclude that the proposed method has better generalization performance than others.

4. CONCLUDING REMARKS

The preliminary results from this ongoing study suggest that the proposed ensemble of crossover operators could improve the generalization performance of RCGA. The current research is still being carried out by considering ensemble of various crossover operators. Further, it is worth noting that a similar ensemble concept can be applied to mutation operators in order to construct robust candidate solutions. The development of an adaptive mutation scheme is especially important because it can significantly enhance the performance of GA [6].

5. ACKNOWLEDGMENTS

This work was supported by the Systems Biology Infrastructure Establishment Grant (2013) provided by Gwangju institute of Science and Technology (GIST), and the Ministry of Culture, Sports and Tourism (MCST) and Korea Creative Content Agency (KOCCA) in the Culture Technology (CT) Research & Development Program 2012.

6. REFERENCES

[1] Wolpert, D. H. and Macready, W. G. 1997. No free lunch theorem for optimization. IEEE Trans. on Evolutionary Computation, 1(1), pp. 67–82.

[2] Herrera, F., Lozano, M., and Sánchez, A. M. 2003. A taxonomy for the crossover operator for real-coded genetic algorithms: an experimental study, Int. J. Intell. Syst. 18, pp. 309–338.

[3] Eshelman, L. J. and Schaffer, J. D. 1993. Real-coded genetic algorithms and interval-schemata. In: Whitley LD, editor. Foundations of Genetic Algorithms 2. San Mateo, CA: Morgan Kaufmann Publishers, pp. 187–202.

[4] Deb, K. and Agrawal, R. B. 1995. Simulated binary crossover for continuous search space. Complex Systems, 9, pp. 115–148.

[5] Michalewicz, Z. 1996. Genetic Algorithms + Data Structures = Evolution Programs, 3rd ed., Springer, New York.

[6] Lobo, G. F., Lima, C. F. and Michalewicz, Z. (Eds) 2007. Parameter setting in evolutionary algorithms. Springer, New York, 2007.