Chaotic Crossover Operator on Genetic Algorithm

Hüseyin Demirci and Ahmet Turan Özcerit
Sakarya University, Computer Engineering, Sakarya, Turkey
Email: {huseyind, aozcerit}@sakarya.edu.tr

Hüseyin Ekiz
Süleymanşah University, İstanbul, Turkey
Email: hekiz@ssu.edu.tr

Akif Kutlu
Computer Engineering, S. Demirel University, Isparta, Turkey
Email: akutlu@hotmail.com

***Abstract*—In this paper, chaos based a new arithmetic crossover operator on the genetic algorithm has been proposed. The most frequent issue for the optimization algorithms is stuck on problem's defined local minimum points and it needs excessive amount of time to escape from them; therefore, these algorithms may never find global minimum points. To avoid and escape from local minimums, a chaotic arithmetic crossover operator has been employed on a genetic algorithm. Unimodal and multi modal benchmark functions have been used for comparing and test procedures. The genetic algorithm with this new arithmetic crossover operator has yielded better results than original arithmetic crossover operator does. With this new chaos based arithmetic crossover operator diversity and uniqueness of the genetic algorithm's late population are increased. Therefore, even in the late stages of the optimization process, the genetic algorithm tried to search the entire search space and improved the best solution.**

***Index Terms*—genetic algorithm, chaos, crossover operator**

I. Introduction

Optimization process is to find the desired solution for a predefined problem. The easiest and fastest way to solve the optimization problem is to employ fast computers and proper algorithms. Besides, in some cases, finding the exact solution can introduce unacceptable amount of time. To overcome such difficulties, heuristic search algorithms have been developed such as Genetic Algorithms (GA), Differential Evolution Algorithms (DEA), Particle Swarm Optimizations (PSO), and Ant Colony Optimizations (ACO). GA is the most popular population based heuristic algorithm since it was developed by Holland in 1975 [1]. This algorithm runs faster and requires less memory. GA individuals have limited shared memories. They can only remember the best individual's variables and result. GA uses this information on the calculation of new individuals. With this feature, GA consumes less memory than other optimization algorithms do and works efficiently. Sometimes, GA may stuck on the local minimums like other algorithms. To solve this downside, some steps of GA can be modified.

In this paper, we have presented a chaos based Chaotic Arithmetic Crossover operator (CAC) to improve the population diversity of GAs. CAC utilizes unpredictability and randomness of the chaos theory. By the help of the chaos theory, GA does not stuck on local minimums and finds near exact solution all the time.

II. Background and Related Work

A. Genetic Algorithm

GA is a population-based algorithm that inspired from biological reproduction on cellular level [1]. In this algorithm, population is acquired from individual genes. GA has five critical steps, which affect the performance of the algorithm, as seen in pseudo code of GA given below.

Algorithm 1	Genetic Algorithm
1:	Create initial population.
2:	Repeat
3:	Evaluate fitness value for each individual.
4:	Select parents according to selection scheme.
5:	Create new children by crossover.
6:	Mutate children.
7:	Until Stopping criteria

In genetic operator part of GA, two important steps affect the performance of the algorithm directly. The first one is crossover step that selects two combined parents regarding to selection scheme to generate a new child, which eliminates poor individuals and helps GA to convergence. The latter one is the mutation step that updates new child's random part.

The steps of GA has have been improved by several studies recently. New crossover operators [2, 3, 4],

Manuscript received April 27, 2015; revised August 20, 2015.

doi: 10.12720/jait.6.4.217-220

selection mechanism [5] and mutation operator [6] have been introduced and reviewed [7, 8, 9, 10].

B. Chaotic Systems

Chaotic systems have been developed from chaos theory, which is a study field in math. The idea is based on the unpredictability aspect of nature. In chaotic systems, even a small change in the inputs can cause a considerable change in the outputs. This situation is called butterfly effect and it has been first proposed in 1880s by Henri Poincaré [11].

In recent studies, chaotic number generators developed from chaotic systems have been used for generating random numbers and they have yielded better results. Chaotic sequences in evolutionary algorithms were proposed in [12] and studies on chaotic sequences were introduced in [13].

III. Proposed Work

In related works, GA algorithm works fast and efficiently, however, like other optimization algorithms, GA stucks on local minimums. The efficiency of GA depends on population diversity and to achieve better diversity, the crossover part of GA must produce unique individuals.

The number generators in almost all compilers are based on seeded structures. Number generators generate identical set of numbers unless varied seeds are used. At the end of GA procedure, gene pool is filled by near solutions. Due to the seed structure of the compiler, selection from these solutions can make new individuals as similar as before and this drawback can eventually leads to stuck on local minimums.

In cellular level, crossover with obtained gene from the parents produces unique individuals even if the parents are always the same. To maintain the uniqueness, one of the parent's gene must be dominant and the other must be recessive. In order to simulate this situation, we used chaos based CAC operators. Mathematical expression of the CAC operator is given in (1).

$$c(x_i) = m * p_1(x_i) + (1-m) * p_2(x_i) \quad (1)$$

where x_i is the new generated child's position in dimension (i), m is the random number generated by chaotic random number generator, $p_1(x_i)$ and $p_2(x_i)$ are the positions of the parent 1 and parent 2 in dimension (i).

We have used logistic map in chaotic number generator because of the map's simplicity. Mathematical expression of the chaotic number generator is given in (2)

$$\begin{aligned} map &= 4 * z * (1-z) \\ m &= (0.5 * r) + (0.5 * map) \end{aligned} \quad (2)$$

In equation (2), z and r are the different random numbers generated from compiler's random number generator and m is the generated random number from chaotic number generator and used in (1).

GA with this new crossover operator has been tested on multi modal and unimodal benchmark functions. We have set population size to 100 and have used Gaussian mutation operator along with roulette-wheel selection scheme. All parameters used are given in Table I.

TABLE I. Parameters of Genetic Algorithm

Parameter	Value
Population Size	100
Selection Scheme	Roulette Wheel
Crossover Rate	0.75
Mutation Operator	Gaussian
Mutation Rate	0.1
Mutation Variance	1.0 decreasing to 0.1

IV. Results

A. Test Problems

In this study, we have employed a unimodal and two multi-modal benchmark functions that are popular and used in several studies [2], [3].

The first one is unimodal and it is called Ellipsoidal function that is given in (3).

$$f_1(x) = \sum_{i=1}^{n} i * x_i \quad (3)$$

The second function is multi modal and called Rastrigin function that is given in (4).

$$f_2(x) = x_i^2 - 10 * \cos(2 * \pi * x_i) + 10 \quad (4)$$

The last multi modal function is Ackley function, which is given in (5).

$$\begin{aligned} f_3(x) = 20 + \exp(1) - 20 * \exp\left(-0.2 * \sqrt{\frac{1}{n} * \sum_{i=1}^{n} x_i^2}\right) \\ - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos(2 * \pi * x_i)\right) \end{aligned} \quad (5)$$

In all functions, n stands for dimension and i denotes the current dimension. Initialization range of the functions is shown in Table II.

TABLE II. Initialization Range of Benchmark Functions

Function	Initialization Range
Ellipsoidal Function	(-100, 100)
Rastrigin Function	(-5.12, 5.12)
Ackley Function	(-32.768, 32.768)

All test runs for 10, 20 and 30 dimensions for each function. GA runs for 1000, 1500 and 2000 iteration in 10, 20 and 30 dimensions respectively. Each function runs for 15 times.

B. Results

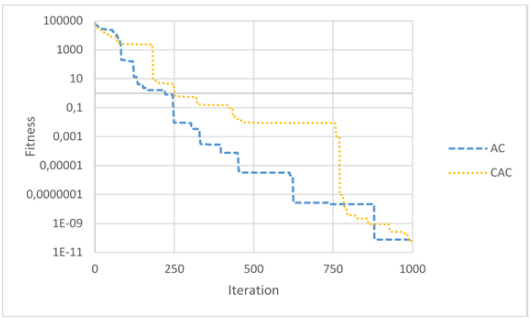
Comparative results of arithmetic crossover and CAC on benchmark functions are given in Table III. GA with AC stucks on local minimums in most cases especially in

higher dimensions. However, GA with CAC never stucks on the local minimums and reached the near global minimum point in every test cases even in higher dimensions.

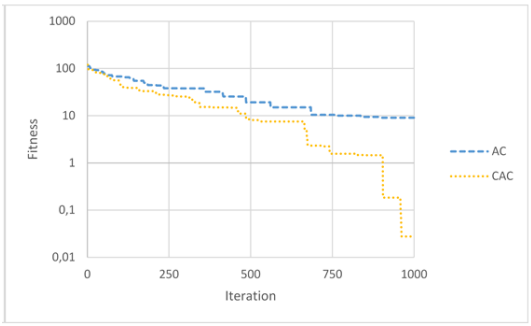
Table III shows that especially in Rastrigin function, GA with CAC algorithm has yielded clearly better results than GA with AC algorithm and only in Ackley function in 30 dimensions; GA with AC yielded slightly better results. Besides, our approach has always outperformed the original GA algorithm. Fig. 1, Fig. 2, and Fig. 3 illustrates the relationship between fitness and iteration when Ellipsoidal, Rastrigin, and Ackley functions used in three varied dimension size i.e. 10, 20, and 30.

TABLE III. TEST RESULTS OF GENETIC ALGORITHM WITH AC AND CAC

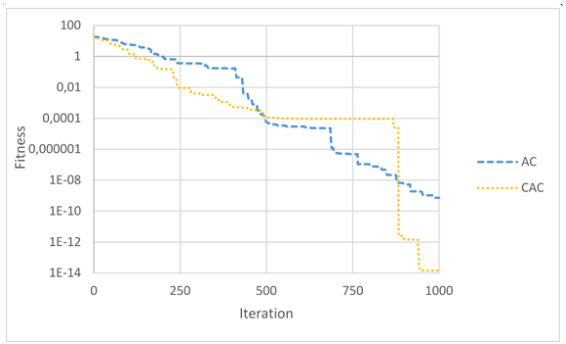
Function	Dimension	GA with AC Mean (Std-dev)	GA with CAC Mean (Std-dev)
Ellipsoidal Function	10	7.77683E-11	5.42E-11
		(2.76868E-10)	(1.03E-10)
	20	2.34379E-29	4.1598E-48
		(7.81952E-29)	(1.55646E-47)
	30	1.14441E-44	4.95971E-94
		(3.02353E-44)	(1.802E-93)
Rastrigin Function	10	9.053261337	0.027726909
		(13.78879335)	(0.101600676)
	20	8.970391741	0.045290179
		(21.95426934)	(0.123722335)
	30	20.05002924	6.68703E-05
		(39.8305619)	(0.000250206)
Ackley Function	10	7.28519E-10	1.44773E-14
		(2.53808E-09)	(5.39451E-14)
	20	4.53021E-12	9.35578E-13
		(1.69474E-11)	(3.50227E-12)
	30	2.16123E-15	2.87178E-15
		(1.57107E-15)	(1.24068E-14)



(a)

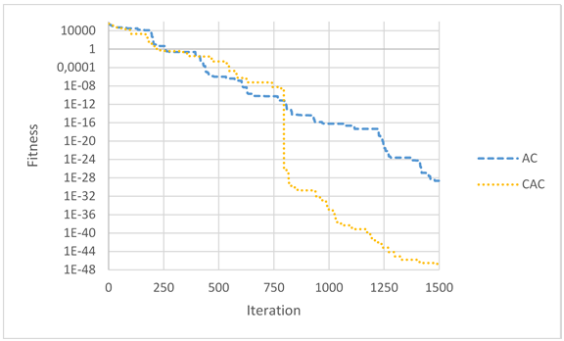


(b)

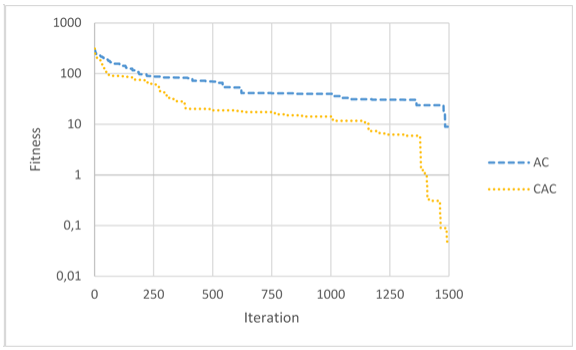


(c)

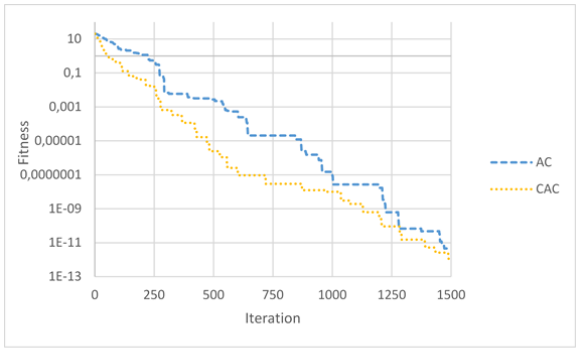
Figure 1. (a) Ellipsoidal Function, (b) Rastrigin Function, (c) Ackley Function at 10 Dimension.



(a)



(b)



(c)

Figure 2. (a) Ellipsoidal Function, (b) Rastrigin Function, (c) Ackley Function at 20 Dimension

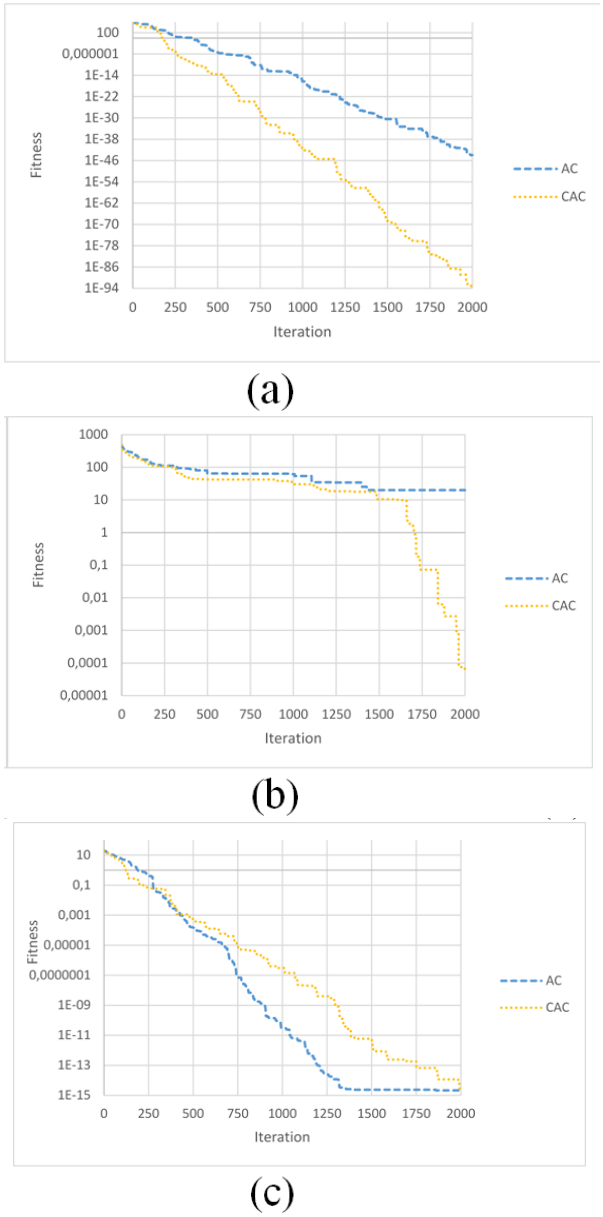


Figure 3. (a) Ellipsoidal Function, (b) Rastrigin Function, (c) Ackley Function at 30 Dimension

V. CONCLUSIONS

GA with CAC algorithm has outperformed GA with AC algorithm. The effect of chaotic systems on the crossover operation leads improved population diversity and uniqueness in the late stages of GA crossover step. Therefore, GA does not stuck on local minimums and probability of the finding near approximate solution has been increased.

For future research, the rest of steps of GA such as mutation can be united with chaotic systems. The effect of the other maps for chaos can be investigated and hybrid stages for GA can be studied.

REFERENCES

[1] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*, U Michigan Press, 1975.

[2] M. J. Varnamkhasti, L. S. Lee, M. R. A. Bakar, and W. J. Leong, "A genetic algorithm with fuzzy crossover operator and probability," *Advances in Operations Research*, vol. 2012, [Online].Available: http://dx.doi.org/10.1155/2012/956498

[3] Y. Kaya, M. Uyar, and R. Tek, "A novel crossover operator for genetic algorithms: Ring crossover," arXiv preprint arXiv: 1105.0355, 2011.

[4] D. Vrajitoru, "Crossover improvement for the genetic algorithm in information retrieval," *Information Processing & Management*, vol. 34, no. 4, pp. 405–415, 1998.

[5] A. Lipowski and D. Lipowska, "Roulette-wheel selection via stochastic acceptance," *Physica A: Statistical Mechanics and its Applications*, vol. 391, no. 6, pp. 2193–2196, 2012.

[6] M. Srinivas and L. M. Patnaik, "Adaptive probabilities of crossover and mutation in genetic algorithms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 24, no. 4, pp. 656–667, 1994.

[7] I. Abuiziah and N. Shakarneh, "A review of genetic algorithm optimization: Operations and applications to water pipeline systems," *International Journal of Mathematical, Computational, Physical and Quantum Engineering*, vol. 7, no. 12, pp. 1283-1289, 2013. [Online]. Available: http://waset.org/Publications?p=84

[8] R. Sivaraj and T. Ravichandran, "A review of selection methods in genetic algorithm," *International Journal Of Engineering Science And Technology*, vol. 3, no. 5, pp. 3792–3797, 2011.

[9] M. R. Noraini and J. Geraghty, "Genetic algorithm performance with different selection strategies in solving tsp," in *Proc. the World Congress on Engineering*, vol. 2, London, UK, 2011.

[10] A. Otman and A. Jaafar, "Article: A comparative study of adaptive crossover operators for genetic algorithms to resolve the traveling salesman problem," *International Journal of Computer Applications*, vol. 31, no. 11, pp. 49–57, October 2011.

[11] H. Poincaré, "Sur le problème des trois corps et les équations de la dynamique," *Acta Mathematica*, vol. 13, pp. 1–270, 1890.

[12] R. Caponetto, L. Fortuna, S. Fazzino, and M. G. Xibilia, "Chaotic sequences to improve the performance of evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 7, USA, pp. 289-304, 2003.

[13] L. J. Yang and T. L. Chen, "Application of chaos in genetic algorithms," *Communications in Theoretical Physics*, vol. 38, U.K., pp. 168-172, 2002.

Hüseyin. Demirci was born in Konya, Turkey, 1990. He received Bachelor Degree in Computer Engg. from Selçuk University in 2012 and his Master Degree in Computer Engineering from Sakarya University in 2014. He is a research assistant in Computer Engineering Department, Sakarya University. His technical interest include soft computing and meta-heuristic algorithms.

Ahmet Turan. Özcerit was born in Istanbul, Turkey, in 1968. He received B.Sc. in Electronics Engg. from Gazi University in Ankara in 1990. He completed his Ph.D. at The University of Sussex in U.K. in 1999. He has been teaching since 2001 at Sakarya University in Turkey as Associate Professor. His research area includes embedded system design, fault-tolerance, wireless sensor networks, steganography and optimization algorithms.

Hüseyin. Ekiz was born in Gaziantep, Turkey, in 1963. He received B.Sc. in Electronics Engg. from Gazi University in Ankara in 1985. He completed his Ph.D. at The University of Sussex in U.K. in 1997. He worked at Sakarya University as full professor between 1997 and 2012. He is vice chancellor of Suleyman Sah University. His research area includes industrial networks, network modelling and programming.

Akif. Kutlu was born in Kocaeli, in Turkey, in 1969. He received B.Sc. in Computer Engg. from Marmara University in Istanbul in 1991. He completed his Ph.D. at The University of Sussex in U.K. in 1998. He has been teaching at Suleyman Demirel University, in Turkey as full professor. His research area includes embedded systems, industrial networks, network modelling and programming.