2010 Fourth International Conference on Genetic and Evolutionary Computing

Continuous CLA-EC

Sakineh zanganeh
Computer Engineering Department
Islamic Azad University-Baft Branch
Baft, Iran
zanganeh_60@yahoo.com

Mohammad Reza Meybodi
Computer & IT Engineering Department
Amirkabir University of Technology
Tehran, Iran
mmeybodi@aut.ac.ir

Meisam Hosseini Sedehi
Computer & IT Engineering Department
Amirkabir University of Technology
Tehran, Iran
Hosseini.Meisam@gmail.com

***Abstract*— Standard CLA-EC which is introduced recently is an evolutionary computing model obtained by combining cellular learning automata (CLA) model and evolutionary computing (EC) model. Some drawbacks of this model are low convergence speed and low accuracy for some optimization problems. In this paper a new version of CLA-EC called Continuous Action Set CLA-EC or in short Continuous CLA-EC is proposed. In this new version, the finite action set learning automaton in each cell is replaced by a continuous action set learning automaton. To show the effectiveness of the proposed model it is tested on some function optimization problems. The results of experimentations have shown that the proposed Continuous CLA-EC comparing to standard CLA-EC has both higher accuracy and speed of convergence.**

Keywords—CLA-EC; Continuous CLA-EC; Evolutionary Algorithm; Cellular Learning Automata; Continuous Action Set Learning Automata; Optimization.

I. INTRODUCTION

Evolutionary Algorithms (EA) are inspired by Darwin's theory of natural evolution. In nature, the better species are evolved by means of natural selection and random variation. EAs follow this approach and simulate the natural selection and variation to evolve a better solution to a problem [1][2]. Evolutionary computation (EC) uses iterative progress, such as growth or development in a population. This population is then selected in a guided random search to achieve the desired end. they are often applied to optimization problems where specialized techniques such as gradient based algorithms, linear programming, dynamic programming are not available or standard methods fail to give reasonable answers due to multimodality, non-differentiability or discontinuity of the problem at hand [3]. Evolutionary computation can be done in parallel. The basic idea behind most parallel EAs (PEA) is to divide their tasks into chunks and then solve the chunks simultaneously using multiple processors. This divide-and-conquer approach can be applied to EAs in many different ways, and the literature contains many examples of successful parallel implementations. There are four main types of parallel EAs: global single-population master-slave EAs, single-population fine-grained PEA, multiple-population coarse-grained EAs and hierarchical combinations [4]. More information about fine-grained PEAs can be found in [5][6][7].

Cellular learning automata (CLA) [8] which is introduced recently is a combination of the cellular automata (CA) [9] and learning automata (LA) [10]. CLA consists of a large number of simple components. The basic idea of CLA is to use learning automata to adjust the state transition probability of stochastic CA. In [11][12] A Fine Grained Evolutionary Algorithm based on Cellular Learning Automata called standard CLA-EC are reported. Standard CLA-EC is obtained by combining cellular learning automata (CLA) model and the evolutionary computing model. More information about Standard CLA-EC can be found in [12].

Some drawbacks of standard CLA-EC are low convergence speed and low accuracy for some optimization problems and using high number learning automata. In this paper a new version of CLA-EC called continuous CLA-EC or in short CCLA-EC is proposed. In this new version, the finite action set learning automaton (FALA) in each cell is replaced by a continuous action set learning automaton (CALA) [13][14][15]. For an r-action FALA, the action probability distribution is represented by an r-dimensional probability vector that is updated by the learning algorithm. For CALA, the action probability distribution is represented by a continuous function and this function is updated by learning algorithm at any stage. To show the effectiveness of continuous CLA-EC it is tested on some function optimization problems. The results of experimentations have shown that the proposed CCLA-EC comparing to standard CLA-EC has both higher accuracy and speed of convergence.

The rest of this paper is organized as follows. Section 2 briefly describes standard CLA-EC. Section 3 introduces the proposed CCLA-EC. The experimental results are given in Section 4. Section 5 is the conclusion.

II. STANDARD CLA-EC

Cellular learning Automata based evolutionary computing (CLA-EC) is obtained by combining cellular learning automata (CLA) model and evolutionary computing (EC) model. This model uses FALA in each cell and hence genome strings have binary representation.

More information about Standard CLA-EC can be found in [11][12].

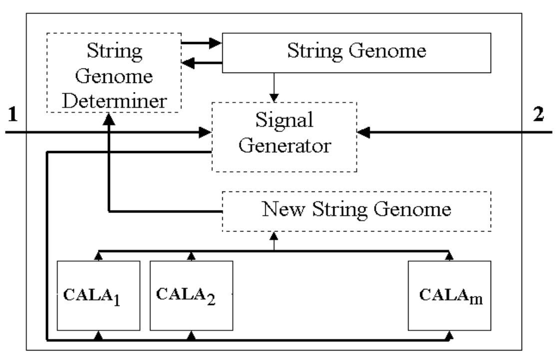
978-0-7695-4281-2/10 $26.00 © 2010 IEEE
DOI 10.1109/ICGEC.2010.53

IEEE computer society

III. THE PROPOSED CONTINUOUS CLA-EC (CCLA-EC)

In this section, a new version of CLA-EC which we call it continuous CLA-EC (CCLA-EC) will be proposed. In continuous CLA-EC unlike standard CLA-EC which uses finite action set learning automaton (FALA) in each cell, it uses continuous action set learning automaton (CALA) and hence genome strings have real representation rather than binary representation. In this model a set of CALA assigned to each cell of CLA model and each genome has two components, model genome and string genome. Model genome is a set of continuous actions learning automata. The set of actions selected by this set of learning automata determines the second component of the genome. In standard CLA-EC model, if the string genome contains *d* gene then a problem with *p* parameters needs *p*d* gene each of which assigned a finite action set learning automata (FALA). Therefore in total for a CLA-EC with *n* cells we need *n*p*d* learning automata whereas in continuous CLA-EC, a problem with *p* parameters needs only *n*p* learning automata. That is, in continuous CLA-EC for encoding each parameter of the problem we need just one CALA whose actions are chosen from real line.

The architecture of a cell of continuous CLA-EC is shown in Fig.1.



1:String Genome of Left Neighborhood
2: String Genome of Right Neighborhood

Figure1. The architecture of a cell in the proposed Continuous CLA-EC

Each cell is equipped with m Continuous action set learning automata. The string genome determiner compares the new string genome with the string genome residing in the cell. The string genome with the higher quality replaces the string genome of the cell. Depending on the neighboring string genomes and the string genome of the cell, a reinforcement signal will be generated by the signal generator.

Continuous CLA-EC model is iterative and the following steps for each cell *i* will be repeated until the termination criterion is met:

1- Every automaton *j* in cell *i* choose an action. That is, it draws sample $\alpha_n^{i,j}$ from normal distribution (*N* $(\mu_n^{i,j}, \sigma_n^{i,j})$).

2- Every Cell *i* generates a new string genome, $new^i = \alpha_n^{i,1}\ \alpha_n^{i,2} \ldots \alpha_n^{i,m}$ which is obtained by concatenating the value of actions of automaton residing in cell *i* at step *n*.

3- Every cell *i* computes the fitness value *f*(.) of string genome new^i; if the fitness of this string genome is better than the one in the cell (ξ_n^i) then the new string genome new^i becomes the string genome of that cell. That is:

$$\xi_{n+1}^i = \begin{cases} \xi_n^i & f(\xi_n^i) \le f(new_{n+1}^i) \\ new_{n+1}^i & f(\xi_n^i) > f(new_{n+1}^i) \end{cases} \qquad (1)$$

4- Se Cells of the neighboring cells of cell i are selected and the set of selected neighbors of cell i be shown with $N_{se}(i)$. This Selection is based on the fitness value of the neighboring cells according to a selection strategy.

5- Based on selected neighboring cells, a reinforcement vector is generated. This vector becomes the input for the set of learning automata associated to the cell. $\beta_n^{i,j}$ is computed as follows, (equation 2)

$$\beta_n^{i,j} = \sum_{l \in N_{se}(i)} \left(\frac{w_I(\xi_n^{l,j} - \mu_n^{i,j})}{\sigma_n^{i,j}}\right) \qquad (2)$$

where $\beta_n^{i,j}$ is the reinforcement signal given to learning automaton j of cell i at step n and the set of selected neighbors of cell i be shown with $N_{se}(i)$. w_I is the normalized evaluation signal by Genome of cell i and $\xi_n^{l,j}$ is the action selected learning automaton j of cell i. $\mu_n^{i,j}$, $\sigma_n^{i,j}$ are mean and variance of normal distribution learning automaton *j* of cell i respectively.

6- The parameters of normal distribution of learning automaton *j* of cell *i* is updated using equation (3)

$$\begin{aligned} \overline{\sigma_{n+1}^l} &= \max \{\ \overline{\sigma_n^l} + \sqrt{q * \overline{\sigma_n^l})} * N(0,1), \sigma_{min}\} \\ \mu_{n+1}^{i,j} &= \mu_n^{i,j} + \beta_n^{i,j} * \sigma_{n+1}^{i,j} \end{aligned} \qquad (3)$$

Where, q is set to 5.

The overall operation of Continuous CLA-EC is summarized in the algorithm of Fig.2.

```
Initialize;
While not done do
    For each cell i in CCLA do in parallel
        Generate a new string genome;
        Evaluate the new string genome;
        If f(new string genome)>f(old string genome) then
            Accept the new string genome;
        End if
        Select Se cells from neighbors of cell i;
        Generate the reinforcement signal vector;
        Update parameters CALAs of cell i;
    End parallel for
End while
```

Figure2. Pseudo code for continuous CLA-EC

IV. Simulation Results

This section presents simulation results for different function optimization problems and then compares the proposed Continuous CLA-EC with Standard CLA-EC and Continuous Genetic Algorithm (GAc). More information about Continuous Genetic Algorithm can be found in [16][17].

A CCLA-EC completely converges when all CALAs residing in all cells converge, i.e. the population (the set of string genomes residing in cells of CCLA-EC) remains unchanged. Each quantity of the reported results is the average taken over 20 runs. The population size (number of cells) varies from 3 to 47 with increments of four. In proposed CCLA-EC, For the sake of convenience in presentation, we use CCLA-EC(CALA(θ_S),r,se,q) for refer to CCLA-EC algorithm with q cells, neighborhood radius r, the number of selected cells Se when using the CALA with distribution parameters θ_S.

The CCLA-EC has a linear topology with wrap around connection. If the radius of neighborhood is one the neighbors of cell i are cell i-1 and cell i+1.

To show the performance of the proposed continuous CLA-EC, we have arranged several benchmark tests for optimization problems. Five benchmark DeJong's functions [18] and the other benchmark test are given in the table1. These functions have been used in order to compare our proposed method with other methods.

A. *CCLA-EC versus Standard CLA-EC and Continuous Genetic Algorithm*

To show the performance of CCLA-EC for real-valued function optimization, we compare it with Standard CLA-EC and GAc. The GAc uses K tournament selection (k=3), arithmetic crossover and Gaussian mutation. Crossover and mutation is applied with probability 0.85 and 0.05 respectively. The parameters of Standard CLA-EC are the same as the parameters used in [12]. Convergence is considered as the termination condition for all algorithms. For CCLA-EC, Se is set to 2 for all test functions with the exception of F3 and set to 3 for function F3. The results of comparisons are reported in Fig.3 through Fig.8. For all functions, continuous CLA-EC has a better result than Standard CLA-EC and GAc. The results for functions F1, F2 and F5 are given in log scale in order to see the difference. The obtained Results in Table2 show High accuracy our proposed Method in Compare with Other Methods in functions Optimization. In total, reported Results indicating the superiority of the proposed algorithm over the Standard CLA-EC and GAc.

V. Conclusion

In this paper, a new version of CLA-EC which we called it continuous CLA-EC (CCLA-EC) proposed. In CCLA-EC, the finite action set learning automaton in each cell is replaced by a continuous action set learning automaton. The results of experimentations have shown that the proposed CCLA-EC comparing to standard CLA-EC and GAc has both higher accuracy and speed of convergence and also continuous CLA-EC needs fewer numbers of learning automata than standard CLA-EC in almost all cases.

References

[1] T. Back, D. B. Fogel, and Z. Michalewicz, "Evolutionary Computation1:Basic Algorithms and Operators," Institute of Physics Publishing Bristol and Philadelphia, 2000.

[2] T. Back, D. B. Fogel, and Z. Michalewicz, "Evolutionary Computation2: Advanced Algorithms and Operators," Institute of Physics Publishing Bristol and Philadelphia, 2000.

[3] D. E. Goldberg, "Genetic Algorithms in Search, Optimization and Machine Learning," Addison-Wesley, New York, 1989.

[4] S. M. Gorges, "Explicit Parallelism of Genetic Algorithms through Population Structures," Parallel Problem Solving from Nature, pp. 150–159, Springer-Verlag, Berlin, 1991.

[5] H. Mühlenbein, "Parallel Genetic Algorithms, Population Genetics and Combinatorial Optimization," Proceedings of the Third International Conference on Genetic Algorithms, pp. 416–421, Morgan Kaufmann, San Mateo, CA, 1989.

[6] S. Baluja, "Structure and Performance of Fine-Grain Parallelism in Genetic Search," Proceedings of the Fifth International Conference on Genetic Algorithms, pp. 155–162, Morgan Kaufmann (San Mateo, CA), 1993.

[7] D. Whitley, "Cellular genetic algorithms," Proceedings of the Fifth International Conference on Genetic Algorithms, San Mateo, CA: Morgan Kaufmann Publishers, pp. 658, 1993.

[8] H. Beigy and M. R. Meybodi, "A Mathematical Framework for Cellular Learning Automata," Advances in Complex Systems, Vol. 7, No. 3 & 4, pp. 295-319, September 2004.

[9] S. Wolfram, "Cellular Automata and Complexity," Perseus Books Group, 1994.

[10] M. A. L. Thathachar and P. S. Sastry, "Varieties of Learning Automata: An Overview," IEEE Transaction on Systems, Man, and Cybernetics-Part B: Cybernetics, Vol. 32, No. 6, pp. 711-722, 2002.

[11] R. Rastegar and M. R. Meybodi, "A New Evolutionary Computing Model based on Cellular Learning Automata," Proceedings of IEEE conference on Cybernetics and Intelligent Systems 2004 (CIS2004), Singapore, December 2004.

[12] R. Rastegar, M. R. Meybodi, and A. Hariri, "A New Fine Grained Evolutionary Algorithm based on Cellular Learning Automata," International Journal of Hybrid Intelligent Systems, IOS Press, Vol. 3, Number 2, pp. 83-98, April 2006.

[13] G. Santharam, P. S. Sastry, and M. A. L. Thathachar, "Continuous Action Set Learning Aautomata for Stochastic Optimization," J. Franklin Inst. 331B (5), pp. 607–628, 1994.

[14] M. N. Howell, G. P. Frost, T. J. Gordon, and Q. H. Wu, "Continuous Action Reinforcement Learning Applied to Vehicle Suspension Control," Mechatronics Vol. 7, No. 3, pp. 263–276, 1997.

[15] H. Beigy and M. R. Meybodi, "A New Continuous Action-set Learning Automaton for Function Optimization," Journal of the Franklin Institute, No. 343, pp. 27–47, 2006.

[16] R. Chelouan, P. A. Siarry, "continuous genetic algorithm designed for the global optimization of multimodal functions," Journal of Heuristics 6(2000), pp.191-213.

[17] S. Tsutsui, D. E. Goldberg, "Search space boundary extension method in real-coded genetic algorithm," Information Sciences 133(2001), pp. 229-247.

[18] K. A. DeJong, "The Analysis of the Behavior of a Class of Genetic Adaptive Systems," Ph.D. Dissertation, University of Michigan, Ann Arbor, 1975.

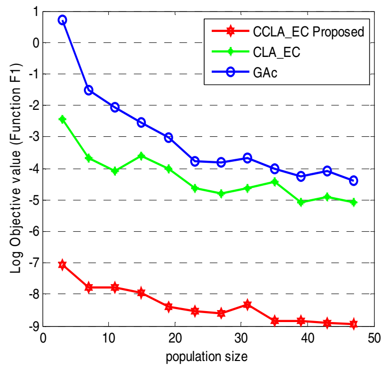


Figure3. Log of Objective value for CCLA-EC, Standard CLA-EC and GAc for Function F1.

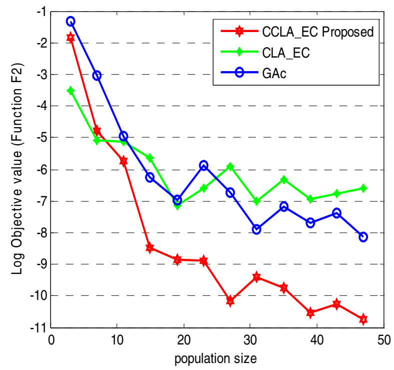


Figure4. Log of Objective value for CCLA-EC, Standard CLA-EC and GAc for Function F2.

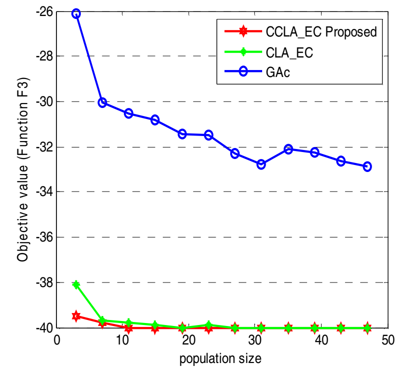


Figure5. Objective value for CCLA-EC, Standard CLA-EC and GAc for Function F3.

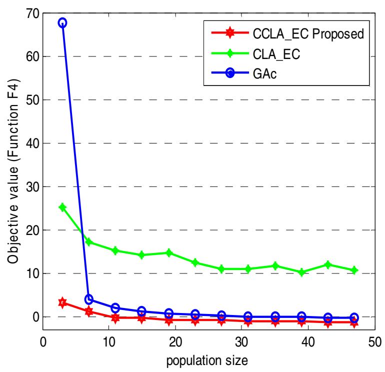


Figure6. Objective value for CCLA-EC, Standard CLA-EC and GAc for Function F4.

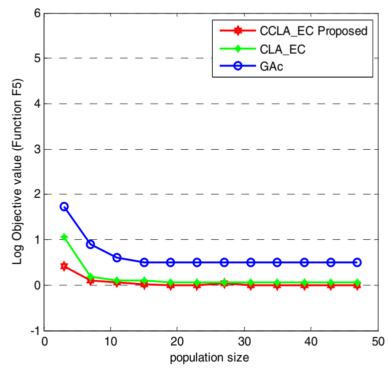


Figure7. Log of Objective value for CCLA-EC, Standard CLA-EC and GAc for Function F5.

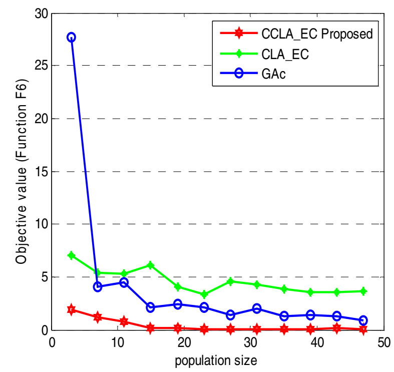


Figure8. Objective value for CCLA-EC, Standard CLA-EC and GAc for Function F6.

TABLE I. TEST FUNCTIONS

Sphere model	$F_1(x) = \sum_{i=1}^{n} x_i^2$	n=3 ,	$-5.12 \le x_i \le 5.12$
Rosenbrock function	$F_2(x) = \sum_{i=1}^{n-1}[100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2]$;	n=2 ,	$-2.048 \le x_i \le 2.048$
Step function	$F_3(x) = \sum_{i=1}^{n} integer(x_i)$	n=5 ,	$-7.12 \le x_i \le 7.12$
Quartic function	$F_4(x) = \sum_{i=1}^{n} ix_i^4 + Gauss(0,1)$	$n = 30$,	$-1.28 \le x_i \le 1.28$
Shekel function	$F_5(x) = (0.002 + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{n}(x_i - a_{ij})^6})^{-1}$	n=2,	$-65.536 \le x_i \le 65.536$
Rastrigin function	$F_6(x) = \sum_{i=1}^{n}[(x_1 - x_i^2)^2 + (x_i - 1)^2]$	n=5,	$-5.12 \le x_i \le 5.12$

TABLE II. OBJECTIVE VALUE

	CCLA-EC proposed	Standard CLA-EC	GAc
F1	9.8606e-009	0.00987	0.0122
F2	3.0792e-009	9.9565e-004	1.9370e-004
F3	-40	-40	-32.9030
F4	-1.8206	10.2014	-1.2790
F5	8.7565e-004	0.0345	1.2354
F6	1.0040e-006	3.2977	0.9017