New Genetic Operator for Dynamic Optimization

Khalid Jebari, Abdelaziz Bouroumi, Aziz Ettouhami

***Abstract*—Recently, Genetic Algorithms (GAs) is interested to multimodal dynamic optimization (MDO). In this kind of optimization, an algorithm requires not only to find the multiple optimal solutions but also to locate a changing optimum dynamically. To enhance the performance of GAs in MDO, our paper proposes a New Genetic Operator (NGO) which called Fertlization. Our approach (NGO) is based on a novel Genetic Algorithms with Dynamic Niche Sharing (GADNS) to find the optimal solutions and to maintain the diversity of the population. An unsupervised fuzzy clustering method is used to track multiple optimums and to perform GADNS. The effectiveness of NGO in dynamic environments, is demonstrated by using Generalized dynamic benchmark generator (GDBG).**

***Index Terms*—Dynamic Niche Sharing, Dynamic Optimization, Fuzzy ClusteringGenetic Algorithms, Unsupervised Learning.**

I. Introduction

TWO challenges must be considered in many optimization problems. The first is their dynamic character and the second is their multimodal aspect. So an optimization algorithms must not only to find the optimal solutions, but also to track the optimal solutions in dynamic environment. For MDO, genetic algorithms (GAs) are an incontestable solutions. GAs are inspired from the Darwinian evolution, which confronted to a dynamic environment in nature. However, when solving MDO, Standard GAs is confronted to a big problem: the convergence. Once GAs converged, they are unable to adapt to the new environment when change occurs. GAs progressively lose diversity as the GAs evolve through generations. In the literature we find several approaches based on GAs to solve this problem of diversity, four types of remedies have been identified [3]:

1) increasing diversity whenever changes are detected, such as increasing mutation rate [7], adjusting the range of mutation [26];
2) spreading out the population, such as using the immigrants approaches [10], [28], or taking into account similarity between individuals [6], [22];
3) the GA is enhanced with memory explicit [27], [17], [4] or implicit [24];
4) using multiple subpopulations[5], [25], [21] approaches.

In this paper, an unsupervised fuzzy clustering method is used in order to identify clusters that correspond to niches. In addition, for each detected cluster (Ci), the algorithm provides the prototypes (Vi), the cluster radius (ri) and size. A spatial separation procedure is used to promote stable sub-populations. Furthermore, as the niching radii and number of peaks are dynamically adjusted, the proposed approach performs the dynamic niche sharing and track the optimal solutions. NGO uses a new genetic operator, adjusts dynamically the selection pressure with modified tournament selection [13] and evolves through three components. The first component is a GA with dynamic niche sharing (GADNS), the second one is an unsupervised fuzzy clustering algorithm, and the third component implements the spatial separation (SS). For maintining also a diversity of population, the fertilization operator is proposed. The principle of this operator is to add new individuals in current population. There new individuals are the prototypes of cluster given by unsupervised fuzzy clustering algorithm. We apply after, mutation and restricted crossover for only those prototypes. This new subpopulation is added to the current population with elitism replacement.

Laboratoire Conception et Systemes;
Faculty of Sciences Mohammed V, Agdal;
University UM5A Rabat Morocco.
E-mail: khalid.jebari@gmail.com

II. Clustering Algorithms

In the absence of information about the distribution of individuals, the fuzzy clustering offers a better possibility of modeling and management of overlapping clusters. So we have opted for a fuzzy classification. Thus, our approach is to introduce a fuzzy clustering based on three phases. The first one is the unsupervised fuzzy learning (UFL) [2] phase that starts by generating the first class which is around the first individual encountered. Then a new cluster created when the current individual presents a small similarity, less than a prefixed threshold Smin, to the entire already existing cluster centers. The second phase is an optimization procedure that ameliorates the learned partition generated during the previous phase. It uses the fuzzy c-means algorithms (FCM)[1]. This clustering is sensitive of the choice of the data similarity, so criterion validity is used in the third phase. This validation (VAL) uses the normalized partition entropy [1] defined as follows:

$$h\left(U\right) = -\frac{1}{\log(c)}\frac{1}{n}\sum_{i=0}^{n}\sum_{j=0}^{c}\left[u_{ij}\log(u_{ij})\right] \qquad (1)$$

Where U: matrix of membership degrees; c:Number of clusters.
More formally, The proposed clustering algorithm UFL-FCM-VAL is described in Algorithm 1 (see APPENDIX A)

III. GAs with Dynamic Niche Sharing

Dynamic niche sharing [19] defines a fixed number of dynamic niches with radii and centers determined by a population sorted. For individuals not in a niche, regular fixed sharing is used. The shared fitness value f_{ds} for an individual within a dynamic niche is:

$$f_{ds,i} = \frac{f_i}{m_{ds,i}} \qquad (2)$$

ISBN: 978-988-19252-1-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

where f_i Fitness value, the dynamic niche count $m_{ds,i}$ is calculated by the fellowing:

$$m_{ds,i} = \begin{cases} n_j & \text{if individual i is within dynamic niche j} \\ m_i & \text{otherwise.} \end{cases} \quad (3)$$

Where n_j is the cardinal of j^{th} dynamic niche;

$$m_i = \sum_{j=1}^{N} sh(d(i,j)) \quad (4)$$

Where N denotes the population size and d(i, j) is a distance measure between the individuals i and j. The sharing function (sh) measures the similarity between two individuals

$$sh(d(i,j)) = \begin{cases} 1 - \left(\frac{d(i,j)}{\sigma_s}\right)^{\alpha} & \text{if d(i,j)} < \sigma_s \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

Where σ_s denotes a threshold of dissimilarity and α is a constant which regulates the shape of the sharing function.
This method requires an estimate of the number of peaks (q) in addition to the niche radius. The primary problem of GADNS is the use of fixed sharing outside the dynamic niches [12]. Note also, the sharing approach has several difficulties. Generally, the number q and the niche radii are often estimated as the maximum number of peaks that could be in the domain, and the minimum niche radius of any optima within the domain, respectively [20].
But, it may be difficult to estimate the number q in the domain. Furthermore, making parameter the same for all individuals means that those peaks are considered as nearly equidistant in the domain [23]
To overcome these limitations, our approach uses a fuzzy clustering technique in order to determine automatically the number of niches q (q=number of clusters given by (UFL-FCM-VAL). Moreover, the radius of each niche is continuously updated.

IV. Description of the proposed approach

The aim is to determine different optima of a dynamic multimodal optmisation using GADNS, the number of peaks q and characteristics corresponding niches (center, radius, cardinal, etc.). The idea is to apply UFC-FCM-VAL at population of solutions produced by GADNS, in order to detect the presence of classes homogeneous and well separated. If the entropy of the fuzzy c-partition obtained [2] is higher than ($>10^{-3}$), these solutions will be again evolved by GADNS and classified by UFC-FCM-VAL. The principle of our technique (figure 1) is based on a three-component. The first component (GADNS) is a GA, which combines dynamic niche sharing, and mating restriction to maintain diversity and to encourage speciation. The second component(UFC-FCM-VAL) is based on an unsupervised fuzzy clustering algorithm, which performs the partition of the individuals given by the first component into a set of C clusters so that each of them corresponds to a niche. The last component (SS) implements the principle of spatial separation to generate sub-populations from the resulting cluster characteristics (center, radius). Hence, individuals undergo a cyclic process throughout the three components of the system. The system is based on the following:

1) each cluster represents a niche;

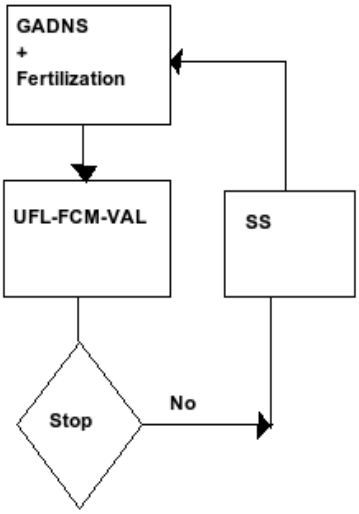

Fig. 1. NGO Structure

2) prototypes represent the individuals of fertilization operator
3) the number of clusters (C) is computed by UFC-FCM-VAL;
4) The subpopulations and their appropriated subspaces are generated using the characteristics (center and radius) of each identified cluster;
5) In order to identify a non-detected niche in the previous cycle, GADNS is used again.

The next sections described the different components .

A. GADNS component

In order to conceive a genetic solution, we have to determine the encoding method. Then the dynamic shared fitness is used for assessing, comparing the solutions and maintaining diversity within population. Hence, starting from an initial population of randomly generated individuals, we evolved this population toward better solutions according to the rules of the selection strategy, crossover and mutation. The details are as follows:
The real encoding scheme was used in the NGO.The initialization of the population of individuals is generated by a random process,the population size is N=200.For the crossover operator, we used Simulated Binary Crossover SBX [8], with crossover rate=0.8. For the mutation, we considered the polynomial mutation [9], with mutation rate starting on 0.2 . To simulate other tests, we also used the Gaussian mutation [18] of adding Gaussian noise to each value of genes that form the chromosome (candidate solution) and the uniform mutation is to select random genes and mutate them, ie, changing their values in the area of the test function. We have used a modified Tournament Selection, which guards in each iteration the best individual. The tournament size is the number of clusters given by UFL-FCM-VAL. The Algorithm 2 summarizes the pseudo code of Tournament Selection Modified (MTS) (APPENDIX B).
This step makes the computation of following for each class or niche i:

- Cardinal of the niche: card (i) = total number of individuals assigned to the niche i

ISBN: 978-988-19252-1-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

- Center (Vi): the mean vector of the cluster elements;

$$V_i = \frac{1}{card(i)} \sum_{j=1}^{card(i)} I_j \quad (6)$$

- radius of the niche

$$\sigma_i = max_{I \in niche}(d(I, v_i)) \quad (7)$$

B. Fertilization Operator

After calcalating and updating the prototypes by UFC-FCM-VAL, the fertilization operator allows to insert new individuals in the population. The first individuals in this sub population are the prototypes F(t), we evolved the individuals by respecting the folowing steps:

- Apply crossover, Let I_f, I_m are two parents, the children are:
  C_1=α*I_f+(1-α)*I_m
  C_2=α*I_m+(1-α)*I_f; Where $\alpha \in$[0,1]
  the result is the sub population F́(t);
- apply Gaussian mutation (the result is the sub population F̈(t));
- each individual in {F(t)U F́(t)U F̈(t)} replaces the nearest individual in the population P(t) if it has a higher fitness.

C. UFC-FCM-VAL component

Once UFC-FCM-VAL() is applied, a defuzzification procedure is performed in order to affect definitely each individual to its natural class for which it presents the maximum membership degree. This results in a final hard c-partition with c cluster centers V_1; V_2; . . . ; V_c, which permit to track the expected optima. In addition to the cluster prototypes, the algorithm also provides some interesting characteristics:

- the number of individuals assigned to the cluster;
- the center V_i and the radius of the cluster r_i;
- the maximum and minimum of inter points similarities within the cluster, S_{max} and S_{min}.

D. SS component

This component permit to affect each individuals to it final class, for which it presents the maximum membership degree. It can be done either from the matrix U^*, by assigning definitely each individual to the class for which he shows the higher degree of membership, or from the matrix V^* by applying the algorithm nearest prototype [14], [2]. The main advantage of this technique is the concretisation of the concept of niche. We note also that SS facilitates the possibility of implementing a local competition, by introducing competition between children and parents of identical niches.

V. EXPERIMENTAL STUDY

The performance of NGO is tested on five problems generated by the benchmark proposed by Li et al [16]. There are seven change types of the system control parameters in the benchmark test. They are small step change, large step change, random change, chaotic change, recurrent change, recurrent change with noise and dimensional change. The framework of the seven change types are described as follows:

- Small step

$$\Delta\phi = \alpha . \|\phi\| . r . \phi_{severity} \quad (8)$$

- Large step

$$\Delta\phi = \|\phi\| . (\alpha . sign(r) + r . (\alpha_{max} - \alpha)) . \phi_{severity} \quad (9)$$

- Random

$$\Delta\phi = N(0,1) . \phi_{severity} \quad (10)$$

- Chaotic

$$\phi(t+1) = A . (\phi(t) - \phi_{min}) . \frac{(1 - (\phi(t) - \phi_{min}))}{\|\phi\|} \quad (11)$$

- REcurrent

$$\phi(t+1) = \phi_{min} + \|\phi\| . \frac{\left(\sin\left(\frac{2\pi}{P} t + \varphi\right) + 1\right)}{2} \quad (12)$$

where $\|\phi\|$ is the change range of ϕ, $\phi_{severity}$ is a constant number that indicates change severity of ϕ, ϕ_{min} is the minimum value of ϕ. $\alpha \in \lceil 0, 1 \rceil$ and $\alpha_{max} \in (0, 1)$ are constant values, which are set to 0.04 and 0.1 in the GDBG system. φ is the initial phase, r is a random number in (-1, 1), sign(x) returns 1 when x is greater than 0, returns 1 when x is less than 0, otherwise, returns 0. The four test problems defined in [15] are: F1 Rotation peak function, F2 Composition of Spheres function,F3 Composition of Rastrigins function, F4 Composition of Griewanks function, F5 Composition of Ackleys function. The parameters of the four problems are set as the same as in [15].

Our first experiment is to investigate the NGOs mechanism of diversity, analyze the fertilization operator. The second experiment, is devoted to the performance of NGO compared with SGA. In the experiments, SGA parameters are as follows: simulated binary crossover [8] , with crossover rate=0.8. For the mutation, we considered the polynomial mutation [9], with mutation rate=0.2. We have used a Tournament Selection with size=4. The population size N =100 For NGO and SGA on a MDO, 30 independent runs were executed with the same set of random seeds. For each run, 50 environmental changes were allowed and the best-of-generation fitness was recorded every generation. The best-of-generation fitness is formulated below:

$$\bar{F}_{BOG} = \frac{1}{G} \sum_{i=1}^{G} \left(\frac{1}{R} \sum_{j=1}^{R} F_{BOG_{i,j}} \right) \quad (13)$$

where G = 50, R = 30 is the total number of runs, and $F_{BOG_{i,j}}$ is the best-of-generation fitness of generation i of run j. The best-of-generation fitness against generation on the MDO with differents rate of fertilization for NGO are showed in Figure 2. First, NGO significantly outperforms SGA; NGO with fertilization give best results than NGO without fertilization. The results of figure 2 show also, that by increasing the rate of individuals introduced to the population through the operator of fertilization, the improved results increases. But more than 0.3 (rate>0.3), the improvement starts to decrease. While using the elitism replacement the performance is high than the others results.

The results of figure 2 validate the benefit of introducing Fertilization, with elitism replacement, into GADNS for MDO. In order to understand the effect of fertilization

ISBN: 978-988-19252-1-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

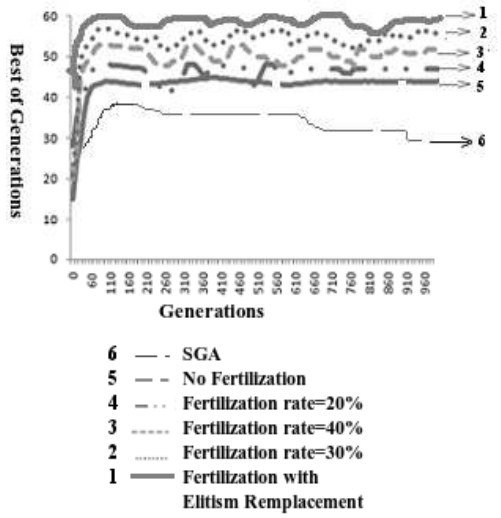

Fig. 2. Dynamic performance of NGO with differents fertilization rates

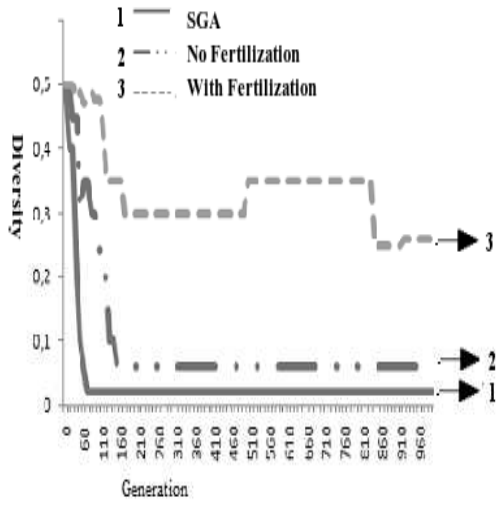

Fig. 3. Diversity Comparison

scheme on the population diversity, we recorded the diversity of the population every generation for each run of a NGO on a MDO. The mean population diversity of a NGO on a MDO at generation over 30 runs is calculated according to the formula:

$$\bar{Div}(t) = \frac{1}{30}\sum_{k=1}^{30}\left(\frac{1}{ln(n-1)}\sum_{i=1}^{n}\sum_{j\neq i}^{n} d_{i,j}(k,t)\right) \quad (14)$$

$D_{i,j}(k,t)$ Euclidean distance between the i-th and j-th individuals at generation t of the k-th run. The diversity dynamics over generation for SGA and NGO with fertilization on MDO is shown in Figure 3. It can be seen that NGO does maintain the highest diversity level in the population while SGA maintains the lowest diversity level. This interesting result shows that approaches that aim at maintaining a high diversity level in the population in dynamic environments with fertilization operator.

In the second experiment we used GDBG system, the parameters of the four problems are set as the same as in [15]. The results show that different problems have a different difficulty for algorithms. F1 is the simplest one to optimize. The composition problems are difficult for algorithms to get the global optima. F3 is the most difficult. Table 1, Table 2 and Table 3 Show that NGO permit to maintan the diversity of population. In all test, HGA performs much better than SGA (Table 4).

VI. Conclusion

In recent years, GAs have been applied to solve MDO with some significants and promising results. For MDO, an optimization algorithm should be able to locate and to track multiple changing optima over time. To solve this problem, researchers have applied several method to enhance the performance of GAs algorithms for MDO. Our paper proposes an unsupervised fuzzy clustering genetic algorithms and the dynamic niche sharing to find the near optimal solutions in promising region and to track multiple optima. A new genetic operator mechanism is introduced to maintain diversity. Fuzzy tournament selection [13] modified is used to adjust dynamically the selection pressure. Generalized dynamic benchmark generator is used to test the performance of the proposed algorithm. The experimental results show the efficiency of the NGO on these test problems. The clustering method used is effective to ameliorate dynamic niche sharing, and fertilization operator introduced maintains much better population diversity. Generally speaking, NGO can effectively locate and track multiple optima in dynamic environments. It would be also interesting to combine other techniques into NGO to further improve its performance in dynamic environments. For example, ants colony to search solution in local region, and NGO to track solutions. An other idea is to use NGO with adaptive crossover and mutation probabilities.

References

[1] J. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum Press, New York, 1981.
[2] A. Bouroumi, and L. M. Essaïdi, Unsupervised fuzzy learning and cluster seeking, Intelligent Data Analysis, vol. 4, no. 3, pp. 241-253, 2000.
[3] J. Branke, Evolutionary Optimization in Dynamic Environments, Kluwer Academic Publishers, Dordrecht, 2001.
[4] J. Branke, Memory enhanced evolutionary algorithms for changing optimization problems, In: Proc. of the (CEC99), IEEE Press, New York 1999, vol. 3, pp. 1875-1882.
[5] J. Branke, T. Kau, L. Schmidt, and H. Schmeck, A multi-population approach to dynamic optimization problems, In: 4th International Conference on Adaptive Computing in Design and Manufacture (ACDM 2000), 2000.
[6] W. Cedeno, V.R. Vemuri, On the use of niching for dynamic landscapes. In: IEEE Int. Conf. on Evolutionary Computation (ICEC97), IEEE Computer Society Press, Los Alamitos 1997, pp. 361-366.
[7] H.G. Cobb, An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous, time-dependent nonstationary environments, Technical Report 6760 (NLR Memorandum), Washington, D.C, 1990.
[8] K. Deb, R.B. Agrawal, Simulated binary crossover for continuous search space, Complex systems, vol. 9, no. 2, pp. 115-148, 1995.
[9] K. Deb, A. Kumar, Real-coded Genetic Algorhithms with Simulated Binary Crossover:Studies on Multimodel and Multiobjective Problems, Complex Systems, vol. 9, no. 6, pp. 431-454, 1995.
[10] J.J. Grefenstette, Genetic algorithms for changing environments. In: 2nd Int. Conf. on Parallel Problem Solving from Nature 1992, pp. 137-144.
[11] A.E. Eiben, Hinterding, R., Michalewicz, Z.: Parameter control in evolutionary algorithms, IEEE Trans. on Evolutionary Computation vol. 3, pp. 124-141, 1999
[12] D. E. Goldberg, L. Wang, Adaptative niching via coevolutionary sharing, In: Quagliarella, D. et al. (Ed.), Genetic algorithms in engineering and computer science, Wiley, Chichester, pp. 21-38, 1997.

ISBN: 978-988-19252-1-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

[13] K. jebari, A. ELmoujahid, A. Dik, A. Bouroumi, A. Ettouhami, Unsupervised fuzzy tournament selection. Applied Mathematical Sciences, vol. 58, pp. 2863-2881, 2011
[14] L.I. Kuncheva, J.C. Bezdek, Nearest prototype classification: clustering, genetic algorithms or random search. IEEE Trans. Systems Man Cybernet, vol. 28, no. 1, pp. 160-164, 1998.
[15] C. Li, S. Yang, T.T. Nguyen, H. Jin, G. Beyer, P.N. Suganthan, Benchmark Generator for CEC2009 Competition on Dynamic Optimization, Technical Report 2008, Department of Computer Science, University of Leicester, U.K, 2008.
[16] C. Li, S. Yang, A generalized approach to construct benchmark problems for dynamic optimization, Proceedings of the 7th Int. Conf. on Simulated Evolution and Learning, Springer, 2008.
[17] S.J. Louis, Z. Xu, Genetic algorithms for open shop scheduling and re-scheduling, In: 11th Int. Conf. on Computers and their Applications (ISCA) 1996, pp. 99-102.
[18] Z. Michalewicz, Genetic algorithms + data structures = evolution programs. Springer-Verlag, London, UK, 1996.
[19] B.L. Miller, M.J. Shaw, Genetic algorithms with dynamic niche sharing for multimodal function optimization, IlliGAL Report no. 95010, 1995.
[20] B.L. Miller, M.J. Shaw, Genetic Algorithms with Dynamic Niche Sharing for Multimodal Function Optimization, IlliGAL Report no. 95010, 1995.
[21] F. Oppacher, M. Wineberg, The shifting balance genetic algorithm: Improving the GA in a dynamic environment, In: Proc. of (GECCO-99), Morgan Kaufman, San Francisco, 1999, pp. 504-510.
[22] J. Rowe, I.R. East, Direct replacement: A genetic algorithm without mutation which avoids deception, In: Evo Workshops, 1994, pp. 41-48.
[23] B. Saäreni, L. Krähenbühl, Fitness sharing and niching methods revisited, IEEE Trans. Evol. Comput. vol. 2, no. 3, pp. 97-106, 1998.
[24] K. Trojanowski, Z. Michalewicz, and J. Xiao, Adding memory the evolutionary planner/navigator, In IEEE Intl. Conference on Evolutionary Compuations, 1997, pp. 483-487.
[25] R.K. Ursem, Multinational GAs: Multimodal optimization techniques in dynamic environments, In: Proc. of (GECCO-2000), Morgan Kaufmann, San Francisco, 2000, pp. 19-26.
[26] F. Vavak, K. Jukes, T.C. Fogarty, Learning the local search range for genetic optimisation in nonstationary environments, In: IEEE Int. Conf. on Evolutionary Computation (ICEC97), IEEE, New York, 1997, pp. 355-360.
[27] S. Yang, Memory-based immigrants for genetic algorithms in dynamic environments, In: Proc. of (GECCO05), ACM Press, New York, 2005, pp. 1115-1122.
[28] S. Yang, Genetic algorithms with elitism-based immigrants for changing optimization problems, Applications of Evolutionary Computing, LNCS 4448, pp. 627-636, 2007.

APPENDIX A
UFL-FCM-VAL()

Data: Individuals: $(I_1, I_2, ...I_N)$
Result: Prototypes $(V_1^*, V_2^*, ..., V_C^*)$; C^*: Number of Clusters
Initialization;
$S_{min} \leftarrow \xi_{min}$;
$h_{min} \leftarrow 1$;
$C^* \leftarrow 2$;
while $S_{min} < \xi_{max}$ **do**
  Apply UFL();
  Apply FCM();
  if $(h_{min} < h())$ **then**
    $h_{min} \leftarrow h()$;
    $C^* \leftarrow C$;
    $V^* \leftarrow V$;
    $U^* \leftarrow U$;
  end
  $S_{min} \leftarrow S_{min} + step$;
end

Algorithm 1: Proposed Clustering Algorithm UFL-FCM-VAL()

APPENDIX B
MTS()

Data: Array: (Random_Table(), Sorted_Table())
Result: Array: (Winner_Table())
Initialization ;
$k \leftarrow C$;
$l \leftarrow 0$;
for $i \leftarrow 0$ **to** k **do**
  Shuffle Random_Table() ;
  $j \leftarrow 0$;
  while $j < N$ **do**
    $C1 \leftarrow Random\_Table(j)$;
    for $m \leftarrow 1$ **to** k **do**
      $C2 \leftarrow Random\_Table(j+m)$;
      if *f(C1)< f(C2)* **then**
        $C1 \leftarrow C2$;
        // f(): Dynamic Shared Fitness
      end
    end
    $j \leftarrow j + k + 1$;
    $Winner\_Table(l) \leftarrow C1$;
    $Winner\_Table(l+1) \leftarrow Sorted\_Table(l)$;
    $l \leftarrow l + 2$;
  end
end

Algorithm 2: Unsupervised Fuzzy Tournament Selection Modified (MTS())

ISBN: 978-988-19252-1-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

TABLE I
ERROR VALUES ACHIEVED FOR PROBLEMS F_1

Technique used	Peaks(m)	Errors	T1	T2	T3	T4	T5
SGA	10	Avg_best	$0.04e^{-4}$	$4.34e^{-5}$	$3.13e^{-4}$	$8.5e^{-4}$	$1.614e^{-4}$
		Avg_worst	40.15	61.18	55.26	69.54	42.15
		Avg_mean	6.15	9.11	12.54	23.48	6.83
		STD	10.45	12.59	14.43	23.89	8.69
	50	Avg_best	$2.04e^{-4}$	$6.14e^{-4}$	$9.53e^{-4}$	$6.35e^{-4}$	$3.41e^{-4}$
		Avg_worst	45.25	48.59	51.69	69.54	34.52
		Avg_mean	8.61	12.16	17.19	19.11	5.48
		STD	10.88	12.58	14.51	20.08	5.38
NGO	10	Avg_best	0	0	0	0	0
		Avg_worst	0.44	19.45	38.63	2.18	15.38
		Avg_mean	0.02	1.81	4.25	0.06	0.89
		STD	0.62	5.23	10.02	0.65	3.26
	50	Avg_best	0	0	0.0000245	0	0
		Avg_worst	0.12	3.23	5.31	1.17	0.43
		Avg_mean	T1	T2	T3	T4	T5
		STD	0.36	3.03	8.51	0.84	0.74

TABLE II
ERROR VALUES ACHIEVED FOR PROBLEMS F_2

Technique used	Errors	T1	T2	T3	T4	T5
SGA	Avg_best	$0.01e^{-4}$	$3.24e^{-5}$	$5.73e^{-4}$	$2.5e^{-4}$	$0.914e^{-4}$
	Avg_worst	98.65	458.12	489.45	123.15	478.54
	Avg_mean	34.15	82.91	128.52	32.54	94.28
	STD	43.68	98.15	31.12	58.12	76.59
NGO	Avg_best	$9.04e^{-5}$	$7.02e^{-6}$	$4.56e^{-5}$	$1.21e^{-6}$	$8.57e^{-6}$
	Avg_worst	19.45	42.05	51.12	9.26	85.69
	Avg_mean	1.32	9.41	10.12	0.23	13.86
	STD	4.17	19.07	18.65	2.13	29.65

TABLE III
ERROR VALUES ACHIEVED FOR PROBLEMS F_3

Technique used	Errors	T1	T2	T3	T4	T5
SGA	Avg_best	$2.69e^{-3}$	$3.43e^{-3}$	$1.18e^{-3}$	$1.85e^{-3}$	$4.84e^{-3}$
	Avg_worst	192.51	345	225.68	127.45	312.14
	Avg_mean	41.87	145.28	187.23	54.65	214.23
	STD	T1	T2	T3	T4	T5
NGO	Avg_best	$7.04e^{-5}$	$9.12e^{-4}$	$3.86e^{-4}$	$1.98e^{-4}$	$4.07e^{-5}$
	Avg_worst	28.59	198.54	145.23	15.28	98.26
	Avg_mean	2.67	47.26	17.52	0.15	19.27
	STD	4.15	81.15	67.84	2.49	86.28

TABLE IV
ALGORITHM OVERALL PERFORMANCE

		F_1(10)	F_1(50)	F_2	F_3
SGA	T1	0.689	0.688	0.569	0.442
	T2	0.648	0.678	0.658	0.345
	T3	0.546	0.548	0.512	0489
	T4	0.323	0.358	0.351	0.289
	T5	0.458	0.523	0.256	0.241
Mark		0.0224685	0.033405	0,056304	0,043344
NGO	T1	0.958	0.978	0.858	0.745
	T2	0.987	0.945	0.745	0.761
	T3	0.845	0.874	0.78	0.421
	T4	0.895	0.812	0.645	0.328
	T5	0.891	0.789	0.532	0.485
Mark		0.06864	0,06567	0,085656	0.06576
Performance: SGA :38 NGO: 71					

ISBN: 978-988-19252-1-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)