Avrupa Bilim ve Teknoloji Dergisi
Özel Sayı 26, S. 383-388, Temmuz 2021

Araştırma Makalesi

www.ejosat.com ISSN:2148-2683

European Journal of Science and Technology
Special Issue 26, pp. 383-388, July 2021

Research Article

Adaptive Genetic Algorithm Renewed by Migration Operator

Rabia Korkmaz Tan[1*], Şebnem Bora[2]

[1] Tekirdağ Namık Kemal University, Çorlu Engineering Fakulty, Department of Cpmputer Engineering, Tekirdağ, Türkiye (ORCID: 0000-0002-3777-2536), rkorkmaz@nku.edu.tr

[2] Ege University, Faculty of Engineering, Department of Cpmputer Engineerin, İzmir, Turkey (ORCID: 0000-0003-0111-4635) sebnem.bora@ege.edu.tr

(International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA) 2021 – 11-13 June 2021)

(**DOI:** 10.31590/ejosat.959683)

ATIF/REFERENCE: Korkmaz Tan, R. & Bora, Ş. (2021). Adaptive Genetic Algorithm Renewed by Migration Operator. *European Journal of Science and Technology*, (26), 383-388.

Abstract

In the present study, the Genetic Algorithm and the developed Adaptive Genetic Algorithm are used to solve optimization problems faced in modeling of complex systems. While using the Genetic Algorithm and the Adaptive Genetic Algorithm, the most important problem encountered is that these algorithms are prone to getting stuck in local bests. For example, in the complex system data optimization process using the Genetic Algorithm, it has been observed that it is frequently fitted to local bests in a certain period of time. The reasons are that, many mutations cannot be performed in the limited number of iterations, and because the number of individuals is limited, the population is filled with the same set of solutions in a short time. Therefore, the migration operator has been added to the Genetic Algorithm and the Adaptive Genetic algorithm in order to avoid local bests and to provide that these algorithms search in a wider area of the large search space of complex systems. In the present study, we have tested these algorithms using the migration operator in the Lotka Volterra Model. When the results are examined, it is observed that the Adaptive Genetic Algorithm with migration operator outperforms the Genetic Algorithm and the targeted success is achieved in complex system optimization. In the rest of the paper, the algorithms used in the Method Section are explained with outlines. In the Experimental Studies Section, the different algorithms are tested and compared with each other in the Lotka-Voltera model and numerical test functions. In the Conclusions and Discussions Section, brief information is given about general results and future studies.

Keywords: Complex Systems, Genetik Algorithm, Adaptive Genetic Algorithm, Migration Operator, Optimization.

Göç Operatörü ile Yenilenen Uyarlanabilir Genetik Algoritma

Öz

Bu çalışmada, karmaşık sistemlerin modellenmesinde karşılaşılan optimizasyon problemlerine yönelik çözüm üretmek amacıyla Genetik Algoritma ve geliştirilen Uyarlanabilir Genetik Algoritma kullanılmıştır. Bu algoritmaları kullanırken karşılaşılan en önemli problem, bu algoritmaların yerel en iyilere takılmasıdır. Örnegin, Genetik Algoritma kullanıldı ğında karmaşık sistem veri optimizasyon işleminde, belirli bir süre içinde yerel en iyilere sıklıkla takıldıgı gözlemlenmiştir. Bunun nedeni, sınırlı sayıdaki iterasyonlarda çok fazla mutasyon gerçekleştirilemeyip birey sayısının sınırlı olmasından dolayı kısa sürede popülasyonun aynı çözüm kümesi ile dolmasıdır. Buradan yola çıkarak Genetik ve Uyarlanabilir Genetik Algoritmalara göç operatörü ekleyerek daha geniş alanda arama yapmaları ve yerel en iyilerden kaçınmaları saglanmıştır. Bu çalışmada Lotka Volterra modeli kullanılarak göç operatörünün kullanıldıgı bu algoritmalar test edilmiştir. Elde edilen sonuçlar incelendiginde göç operatörünün eklendigi Uyarlanabilir Genetik Algoritmanın Genetik Algoritmadan çok daha başarılı oldugu gözlemlenmiş olup, karmaşık sistem optimizasyonunda hedeflenen başarı yakalanmıştır. Çalışmanın devamında Metot bölümünde kullanılan algoritmalar ana hatları ile açıklanmaktadır. Deneysel sonuçlar bölümünde kullanılan farklı algoritmalar lotka-voltera modelinde ve nümerik test fonksiyonların test edilip birbirleri ile karşılaştırılmaktadır. Sonuç ve tartışma bölümünde genel sonuçlar ve gelecekte yapılabilecek çalışmalar hakkında kısaca bilgi verilmektedir.

Anahtar Kelimeler: Karmaşık Sisteler, Genetik Algoritma, Uyarlanabilir Genetik Algoritma, Göç Operatörü, Optimizasyon.

[*] Corresponding Author: Tekirdağ Namık Kemal Üniversitesi, Çorlu Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, İzmir, Türkiye, ORCID: 0000-0002--3777-2536, rkorkmaz@nku.edu.tr

1. Introduction

Complex Systems is a system in which large networks of components with lack of centralized control and simple behaviours' rules lead to complex collective behavior, sophisticated information processing, and adaptation to the changes in the environment. In complex systems, components of the system produce complex behaviours in hard to predict ways and these macroscopic behaviors of the system are called emergent behaviours.

Modeling and simulation techniques are mostly used to facilitate the examination of these systems, which are difficult and costly to examine in the real environment(Di Marzo et al., 2011; Kaddoum, and George, 2012, Guivarch, 2012). Growth of the size of complex systems and possession of large data spaces make the desired emergent behavior hard to appear in the simulation environment. However, only modeling and simulation is not capable of the optimization processes of data space of complex systems (Korkmaz Tan & Bora, 2017a).Therefore, there is a need of a system to optimize these data spaces. Unfortunately, classical optimization methods are not capable of optimizing large and non-linear data spaces of complex systems . Because of this today, meta heuristic algorithms (MHA) are often used in the solution of optimization problems. It has been supported by studies that MHAs are the most appropriate methods that can be used to solve optimization problems of complex systems in modeling and simulation (Calvez, 2007; Salwata, 2010).

Genetics Algorithm (GA) frequently used in optimization processes, has been used in this study. GA is a meta heuristic search algorithm inspired by the process of natural selection and genetics using mutation and crossover methods; thus, it aims to provide a solution to the optimization problem in modeling and simulation of complex systems (Imbault, 2004). The use of GA was preferred in this study, as it searches large data space much faster than classical methods and successful results are achieved, especially when the solution area is large, discontinuous and complex. However, some of GA's problems were encountered during the study and by solving these problems, GA optimization has been made into a more successful algorithm compared to the original algorithm. The problem first encountered was that GA often converged to local bests. In order to solve this problem, the migration operator has been added to the GA. Thus, the problem of getting stuck in the local best was solved while searching in the large search space.

Second problem is the necessity of adapting the strategic parameters of GA to the problem. Optimal strategic parameter values are critical in finding optimum or near-optimum results by the algorithm. Therefore, GA's strategic parameter variables need to be adapted to the problem in this study since getting strategic parameter values to solving the problem improves the performance of the algorithm and enables it to reach the most appropriate solution. In this study, strategic parameters of GA were adjusted using the iterative F-Race algorithm (Birattari et al., 2010). Thus, Adaptive Genetic Algorithm (AGA) was developed, whose strategic parameters are adapted to the problem in this study.

2. Material and Method

2.1. Genetic Algorithm

Holland's GA aims to find the best solution or a result close to the best solution intuitively by using the genetic coding structure of living creatures. Searching large search spaces using classical methods takes a long time and increases the cost. A genetic algorithm can find a solution that is sufficient to meet the expectation in a short time. GA is especially successful in optimization for multidimensional large data spaces of complex systems and GA finds the best solution without getting stuck in local best according to the principle of survival of the best in complex, wide, discontinuous, and multidimensional data space (Korkmaz Tan & Bora, 2017b).

Each of the GA parameters is represented as a gene in biology, whereas the parameter set represents chromosomes. Each chromosome of GAs provides a solution set, these solution sets come together and they form populations.The fitness value of chromosomes, is maximized or minimized depending on the problem and within certain rules. The new generations are obtained by gene exchange of solution sets with the best fitness in the first population. Genetic algorithms include selection, crossover, mutation, and migration operators examined in this study (Korkmaz Tan & Bora, 2017a).

2.2. Genetic Algorirhm Operators

2.1.1. Selection Operator

Selection allows new solution sets (i.e. individuals) with crossover techniques by grouping Selected Solution sets in pairs (Fığlalı and Engin, 2010). In this study, roulette wheel and tournament method were used.

2.1.2. Crossover Operator

Crossover operator is the operator used to produce better solution sets compared to solution sets with the best fitness value. Cross match is applied to solution sets thrown into the pool (Jang, 1997). One-point and two-point crossover methods are used in the study.

2.1.3. Mutation Operator

Due to the principle of conservation of good individuals in GA over time, the gene sequence of individuals are similar to each other and, thus; it negatively affects diversity of gene sequence of individuals, i.e., the searhing process in the large data space. Therefore,, new individuals with reduced diversity in other words, new solution sets are created.

2.1.4. Migration Operator

In order to enable GA to search in a wider area of the large search space of complex systems, the migration operator was used in this study. When using original GA, it is observed that complex system model parameters are prone to getting stuck in local bests within a specified time in the tuning process.The reasons are that, many mutations cannot be performed in the limited number of iterations, and because the number of individuals is limited, the population is filled with the same set of solutions by processing 10 solution sets originally produced at the initial. It's a problem that shows the algorithm is fitted to the

local best. Therefore, the migration operator has been added to the Genetic Algorithm and the Adaptive Genetic algorithm in order to avoid local bests.

Although the use of the migration method is slightly different from the parallel genetic algorithm (Korkmaz Tan & Bora, 2019, 2020; Rebaudengo and reorda, 1992), this study shows that the use of migration operator enables to search in a wider area of the large search space of complex systems and it provides better solutions to the problem caused by stucking at local bests compared to the genetic algorithm and parallel genetic algorithm. Migration operator removes an individual with the poor fitness value from the population by the reverse roulette wheel method (the worst individuals most likely to be selected) and it generates a random set of solutions and includes them in the population. Migration is adjusted to be performed once in each iteration.

2.3. Iterated F-Race Algorithm

Iterated F-Race Algorithm is also seen as a simple form of a sequential model-based method since it is a multiple-phase method that exemplifies a distribution, F-Race algorithm starts working with random parameters for problem examples and F-Race erases the worst values according to statistical tests that it uses as if it is in a race. In the iterative F-Race algorithm, this process continues as cycles during a defined number of iterations and in each cycle new parameter sets are determined by using best candidate parameter values generated in the previous step. Critical parameter values found by Iterarive F-Race algorithm are given to the algorithm. Then, the algorithm is run for adjusting model parameters.

If the fitness value is improved, this critical parameter values are added to the critical parameter array created to be given to the F-Race. The success of adjusting the model parameters of the algorithm is tested by manually giving the algorithm the best critical parameter values found when maximum iteration is reached.

2.4. Adaptive Genetic Algorithm (AGA)

The genetic algorithm has some strategic parameter variables. Some of these variables are genetic crossover method, parental selection percentage, mutation rate, and parental selection method. These parameters are required to be defined in the genetic algorithm and they can take different values in each problem. It is quite difficult to manually determine the best values for each problem. In this study, the necessity of adapting the strategic parameter values of GAs to the problem is discussed. Because finding specific solutions to the problem in which the GA algorithm is used, depends on the strategic parameter values it has. Taking the values of the strategic parameter of GAs suitable for the solution of the problem directly affects the performance of the algorithm and it ensures that the most appropriate solution for the problem is reached. GA is an algorithm developed independent of the problem. It has been observed that when GA is successful in solving some problems, it has not achieved the same success in solving different problems with the same strategic parameter values . Therefore, GA should be adaptable to the problem where it is used in. In order to avoid local bests and increase the speed of approaching the optimum solution, these values should be updated in line with the feedback while the algorithm is running. Strategic parameter values of GAs are set using the Iterated F-Race algorithm, and the GA algorithm has been transformed into an algorithm adaptable to the problem. The pseudo code of the Adaptive Genetic Algorithm (AGA) specially developed for the problem is given below.

Adaptive Genetic Algorithm

```
for pop=1 to maxPop        /*for all the individuals in the population*/
    for par=1 to maxPar do /*maxpar: maximum number of parameters*/
        randomly identify genes in individuals' chromosomes
    end for
        computing the fitness values of solution sets.
end for
do
if fraceCondition=false then
    pArray←par   /*cross, parentSelection, parentSelectionPercent, mutation*/
  /* all the critical values are taken from the manually specified parameter.xml fil*/
else
    pDizi←F-Race /*cross, parentSelection, parentSelectionPercent, mutation*/
    /*all critical values are determined by the frace algorithm*/
end if
for i=0 to parentSelectionPercent
    select 2 individuals by selection method
    make a crossing over between 2 individuals
    apply mutation condition
    apply migration methods
end for
    iteration=iteration + 1
while (iteration < =maxIteration)
```

3. Experimental Studies

3.1. Applied Model

In order to test the proposed approach, the model of the ecosystem of wolf and sheep nemely, Lotka-Voltera Model was examined. The parameters that affect the sustainability of this ecosystem were determined and those parameters were adjusted according to the objectives of the ecosystem. There are three agents in the model: wolf, sheep and grass. The wolf and sheep agents move in a randomized fashion. Both the wolf and sheep agents are in need of energy in order to move in the environment. Because of this, both of them are initialized with a certain level of energy. As the wolf and sheep agents move around the environment, they consume a predetermined amount of energy. If in any case, the energy levels of these two agents drop to zero, they cease to exist. In order to increase their energy levels with the purpose of survival, a wolf hunts sheep and a sheep consumes grass. To further illustrate, if a wolf agent meets a sheep agent, the same wolf agent consumes the sheep agent and acquires energy in accordance with a predetermined parameter value *wolfgainfromfood*. In parallel with a wolf agent, if a sheep agent meets with a living grass agent, just like a wolf consumes sheeps, the sheep agent consumes the grass agent, acquiring a predetermined amount of energy in accordance with the parameter value *sheepgainfromfood*. At the beginning of the simulation, the grass agent is initialized to every grid and they grow at a rate parameter of *grassregrowtime*. If the grass gets consumed by the sheep agent, in time, it regenerates. The life expectancy of the grass agent is randomized, meaning after a

certain amount of random time, it ceases to exist and regenerates again (Korkmaz Tan & Bora, 2019).

3.2. GA Optimization Results

The parameter values after the parameter optimization operation in the Lotka-Voltera model are presented in Table 1. The Model is run 20 times. From this table, it can be observed that after the 10th iteration all the parameter values converge to similar values. From this observation, it is concluded that the Genetic Algorithm is getting stucked at the local bests. Therefore, to achieve a wider search, a migration operator was added to the GA . Later, the Lotka-Voltera model is iteratively run for 20 times in total in order to provide parameter optimization by GA with a migration operator.

Table 1. Solution Sets of Lotka-Volterra Model Dataset Optimized Using GA.

İteration	Individual	Sheep Count	Wolf Count	Sheep Energy	Wolf Energy	Sheep Reproduction	Wolf Reproduction	Grass Regrow	Fitness
8	9	82	68	7,10	18,2	4,7	3,9	41	0,676
8	10	82	68	7,10	22,6	4,7	3,7	24	0,199
9	1	82	68	7,10	18,2	4,7	4,8	41	0,709
9	2	82	68	7,10	18,2	4,7	3,9	41	0,737
9	3	82	68	7,10	18,2	4,7	4,8	24	0,208
9	4	82	68	7,10	18,2	4,7	4,8	41	0,478
9	5	82	68	7,10	18,2	4,7	3,9	41	0,757
10	1	82	68	7,10	18,2	4,7	3,9	41	0,494
10	2	82	68	7,10	18,2	4,7	3,9	41	0,737
10	3	82	68	7,10	18,2	4,7	4,8	41	0,504
10	4	82	68	7,10	18,2	4,7	4,8	24	0,718
10	5	82	68	7,10	18,2	4,7	4,8	41	0,155
11	1	82	68	7,10	18,2	4,7	4,8	41	0,731
11	2	82	68	7,10	18,2	4,7	4,8	41	0,766
11	3	82	68	7,10	18,2	4,7	4,8	41	0,23
11	4	82	68	7,10	18,2	4,7	4,8	41	0,758
11	5	82	68	7,10	18,2	4,7	4,8	41	0,489
12	1	82	68	7,10	18,2	4,7	4,8	41	0,701
12	2	82	68	7,10	18,2	4,7	4,8	41	0,498
12	3	82	68	7,10	18,2	4,7	4,8	41	0,744
12	4	82	68	7,10	18,2	4,7	4,8	41	0,721
12	5	82	68	7,10	18,2	4,7	4,8	41	0,724
13	1	82	68	7,10	18,2	4,7	4,8	41	0,796
13	2	82	68	7,10	18,2	4,7	4,8	41	0,784

In every iteration, 10 solution sets are singularly run for the model and, for each solution set, a fitness value is calculated. When Table 2 is evaluated, it is seen that the GA algorithm with the migration operator by producing more diverse solution sets, finds better solution sets in a wider data space until the end of 20th iteration. Considering the parameter adjusting operation, it is observable that different parameters are obtained until the end of the last iteration. Also, we can observe the evaluation of optimization of GA and AGA by running 10 times and taking the average of the solution set in Table 3. For the minimization operation in GA, the best fitness value is determined to be 0 while worst acceptance value is to be 1. When GA and AGA are compared, we can see that AGA comes up with better solutions as demonstrated in the fitness value and improvement value.

3.3. Parameter Adjusting in Modeling And Simulation

Complex systems tend to be modelled in agent based modelling. The algorithm that is developed is tested in Lotka-volterra model which is also a complex system model. In this study, the Lotka-volterra dataset are optimized by using the GA and AGA and successes of GA and AGA are compared.

Table 2. The Result Of The Ga With Migration Operator Run in The Lotka Voltera Simulation in The Parameter Optimization Problem

Generation (Iteration)	Individual	Sheep Count	Wolf Count	Sheep Energy	Wolf Energy	Sheep Reproduction	Wolf Reproduction	Grass Regrow	Fitness
16	7	135	68	6,9	17,2	7,5	9,7	44	0,441
16	8	135	68	6,9	19,2	7,5	3,4	44	0,718
16	9	135	68	6,2	17,2	7,5	3,4	44	0,812
16	10	135	68	6,9	17,2	7,5	12,8	44	0,394
17	1	135	68	6,9	17,2	7,5	3,4	44	0,896
17	2	135	68	6,9	17,2	7,5	3,4	58	0,317
17	3	135	68	6,9	17,2	7,5	12	44	0,429
17	4	135	68	6,2	32,7	7,5	3,4	44	0,3
17	5	135	68	6,9	17,2	11	3,4	44	0,753
18	1	135	68	6,9	17,2	7,5	3,4	44	0,896
18	2	232	68	6,9	19,9	7,5	3,4	44	0,688
18	3	136	68	6,9	17,2	14,5	3,4	44	0,3
18	4	135	68	6,9	37,3	7,5	3,4	44	0,1
18	5	183	68	6,9	17,2	7,5	3,4	44	0,812
19	1	135	68	6,9	17,2	7,5	3,4	44	0,896
19	2	135	68	6,9	28	7,5	3,4	44	0,3
19	3	330	68	6,9	19,9	9,1	3,4	44	0,812
19	4	135	68	6,9	17,2	6,5	3,4	44	0,898
19	5	183	246	6,9	17,2	7,5	3,4	44	0,306
20	1	135	68	6,9	17,2	7,5	3,4	44	0,896
20	2	159	68	6,9	17,2	9,1	3,4	44	0,606
20	3	135	68	6,9	17,2	6,5	3,4	51	0,447
20	4	135	68	6,9	17,2	6,5	3,4	44	0,598
20	5	135	68	6,9	28	6,5	3,4	22	0,1

To optimize Lotka-voltera dataset, the algorithm is run 10 times and each run is performed in 20 iterations. For each iteration, the number of cycles symbolizing the step time is set to 2000. The averages of results achieved from the algorithm are given in the Table 3. The main aim of the Lotka-voltera model is survival of the species. In this study, the Genetic Algorithm with migration operator (migration-GA) and the Adaptive Genetic Algorithm with migration operator (migration-AGA) were developed to find out the best parameter set that achieves the survival of the species. When looked at the number of cycles in Table 3, migration-GA is shown to end quicker than migration-AGA. The reason why the migration-GA ends quicker is that the species goes extinct quicker when a given solution set is applied to the model. Also, as the fitness value closer to zero, it is shown that the solution set comes closer to the ideal solution set where the solution set is more prone for survival of species. So, given that the program works 10 times and for each it iterates 20 times, migration-AGA have better fitness values in consideration of average fitness values.

Initial best fitness value is the fitness value obtained from the first solution set (parameter set) in the first iteration. Given that the model run 10 times and for each it iterates 20 times, the best fitness value is the value obtained by taking the average of all fitness values during the model running 10 times. It is seen that migration-AGA have better fitness values in consideration of average fitness values. Improvement difference gives the difference between Initial best fitness and best fitness values. Improvement value is the the average of the number of improvements in all the iterations and it is given as the average of all improvement values obtained from each run in the table.

When the table is examined, it is seen from the results obtained that Migration-AGA makes much more improvement than Migration-GA . In this case, Migration-AGA which adapts its parameters to the problem is more successful than Migration-GA algorithm in solving complex systems.

Table 3. Results of GA and AGA Algorithms Developed with The Migration Operator Used to Optimize The Lotka-Volterra Model Dataset.

Algorithm	Step Time	Average fitness value	Initial best fitness value	Best fitness value	Improvement difference	Average local fitness value	Average global fitness value	Improvement Number
Migration-GA	11441	0,799753	0,8555	0,183812	0,624988	0,311543	0,311543	2
Migration-AGA	155234,22	0,366663	0,284967	0,030995	0,207272	0,099049	0,091935	4,6

4. Results and Discussion

In this study, the success of the Genetic Algorithm on data set optimization process in modeling and simulation of complex systems was tested on Lotka Volterra model, which was a complex system model. In the complex system's data optimization process using the Genetic Algorithm, it has been observed that it is frequently stucked at local bests in a certain period of time. In order to solve that problem, the mutation operator was added to the Genetic Algorithm.

As results are examined it is seen that adding the mutant operator to the genetic algorithms provides that the search is performed in a wider area of the large search space of complex systems and the solution sets have better fitness values.

Also, the Adaptive Genetic Algorithm we developed has the ability to be adapted to problems. Success of the Migration-AGA and Migration-GA algorithms to which migration operator was added were tested on the Lotka-volterra model and their performance were compared. In the direction of the results obtained from the tests at the end of the study. Migration-AGA provides much better solution sets compared to the other algorithms examined in this study.

Referances

Birattari, M., Yuan, Z. Balaprakash, P., & Stützle, T. (2010). *Experimental Methods for the Analysis of Optimization Algorithms*, Springer-Verlag Berlin Heidelberg.

Calvez B., & Hutzler G. 2007. Adaptive Dichotomic Optimization: a New Method for the Calibration Of Agent Based Models, *21st Annual European Simulation and Modelling Conference (ESM 2007)*, Malta, 415-419.

Di Marzo Serugendo, G., Gleizes, M.P., & Karageorgos, A. (2011). *Self-organising software from natural to artificial adaptation*, first editör, Heidelberg, Berlin: Springer-Verlag.

Fığlalı A., & Engin O. (2002). Genetik algoritmalarla akış tipi çizelgelemede üreme yöntemi optimizasyonu, *İTÜ Dergisi*, 1-6.

Guivarch, V., Camps, V., & Peninou, A. (2012). AMADEUS: an adaptive multi-agent system to learn a user's recurring actions in ambient systems, *Advances in Distributed Computing and Artificial Intelligence Journal,* 3 (1): 1-10.

Imbault, F., & Lebart, K. (2004). A stochastic optimization approach for parameter tuning of support vector machines, Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR, Cambridge, 597- 600.

Jang, J.S.R. (1997). *Neuro-Fuzzy and soft computing, A Computational Approach To Learning and Machine Intelligence*, Derivative- Free Optimization, Prentice-Hall, USA.

Kaddoum, E. & George, J.P. (2012). Collective self-tuning for complex product design (short paper), *in IEEE International Conference on Self- Adaptive and Self-Organizing Systems(SASO)*, Lyon, CPS, (electronic medium).

Korkmaz Tan, R., & Bora, ¸ S. (2017a). Parameter tuning algorithms in modeling and simulation, *International Journal Of Engineering Science and Application*, 1 (2): 58-66.

Korkmaz Tan, R. & Bora, ¸ S. (2017b). Parameter tuning of complex systems modeled in agent based modeling and simulation, *World Academy of Science, Engineering and Technology International Journal of Computer and Information Engineering*, 11 (12): 1301-1310.

Korkmaz Tan, R., & Bora, Ş. (2019). Adaptive parameter tuning for agent-based modeling and simulation. *Simulation: Transactions of the Society for Modeling and Simulation International 2019*; 95 (9): 771–796.

Korkmaz Tan, R., & Bora Ş.,(2020). Adaptive modified artificial bee colony algorithms (AMABC) for optimization of complex systems, *Turkish Journal of Electrical Engineering & Computer Sciences,* 28 (5): 2602-2629

Rebaudengo, M., & Reorda, M.S. (1992). An experimental analysis of effects of migration in parallel genetic algorithms, *EWPDP93:IEEE/Euromicro Workshop on Parallel and Distributed Processing, Gran Canaria (E), Gennaio*, 232-238.

Salwala, C. Kotrajaras, V. & Horkaew, P. (2010). Improving performance for emergent environments parameter tuning and simulation in games using GPU*, Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on*, 2: 37-41.