ISSN: 1803-3814

SOMA NETWORK MODEL BASED ON NATIVE VISIBILITY GRAPH

Lukas Tomaszek, Ivan Zelinka

VSB - Technical University of Ostrava
Department of Computer Science, FEI
Tr. 17. Listopadu 15, Ostrava
Czech Republic
lukas.tomaszek@vsb.cz, ivan.zelinka@vsb.cz

Abstract: *In this article, we want to propose a new model of the network for analyzing the evolution algorithms. We focus on the graph called native visibility graph. We show how we can get a time series from the run of the self-organizing migrating algorithm and how we can convert these series into a network. At the end of the article, we focus on some basic network properties and we propose how can we use these properties for later investigation. All experiments run on well-known CEC 2016 benchmarks.*

Keywords: *complex networks, evolution algorithms, self-organizing migrating algorithm, native visibility graph, time series*

1 Introduction

Evolution algorithms are numerical algorithms, which are based on Darwin's and Mendel's theory of evolution. The main idea is the transfer of the parental genome to new offsprings and the subsequent release of the living space. The most common algorithms, based on this theory, are genetic algorithms [5], differential evolution (DE) [13], and self-organizing migrating algorithm (SOMA) [4, 16].

In our research, we are attempting to convert a run of evolution algorithm into a network. Then, we can analyze the network and get some information about the algorithm. According to it, we can improve the algorithm, to be better and faster. We can see the visualization of the whole process in Figure 1. Also, the main ideas have been described in articles [17, 18].

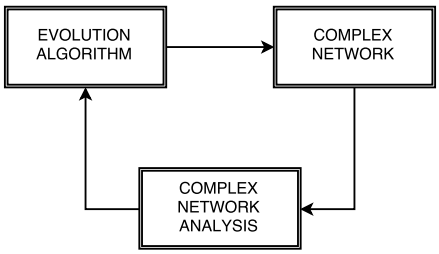


Figure 1: Motivation

Till now, all experiments focused on networks, where edges represented interactions between individuals in the population. E.g., crossing between individuals in the population in the DE [12, 11], interactions between the leader and all other individuals in the SOMA [19, 15], or interactions between fireflies in the firefly algorithm [7].

The main aim of this article is to propose a different possibility of capturing the inner dynamics of the algorithm into a network. Also, we want to count some basic network properties for a better network description and propose how we can use these properties. In this article, we will focus on the graph called native visibility graph (NVG) [8] and SOMA [4, 16].

In the next parts, we firstly describe NVG, SOMA and CEC 2014 benchmark functions, after that we show how we can create time series from SOMA and how we can convert these series into the networks, and at the end of this article we make some basic analysis on these networks and we propose how we can use these results.

2 Native visibility graph

NVG allows us to convert any time series into a network. In such net, each time value represents one vertex. There is a link between two vertices when all time values between these ones lies below their connecting line. In other words, we can imagine that we connect by edge. Value at time t_i will be connected all adjacent values with value at time t_{i+1}. Such connections create a landscape, and we add an edge between two values (vertices) if we are able to see directly from the first one to the second one.

More formally, according to [8], we can establish the following visibility criteria: two arbitrary data values $(t_i, f(t_i))$ and $(t_j, f(t_j))$ will have visibility, and consequently will become two connected vertices in the graph, if any other data $(t_k, f(t_k))$ placed between them fulfills (1). Example of such conversion, we can see in Figure 2.

$$f(t_k) < f(t_j) + (f(t_i) - f(t_j)) \frac{t_j - t_k}{t_j + t_i} \tag{1}$$

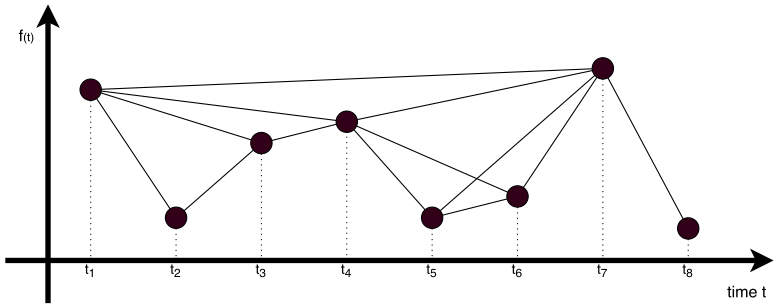


Figure 2: Natural visibility graph - conversion

3 Self-organizing migrating algorithm

SOMA is a stochastic optimization algorithm based on the social behavior of competitive-cooperating individuals [4, 16]. Before the start of the algorithm, we have to set up the parameters and we have to define a cost function. We can see all required parameters, also with recommended range and used setup for our experiments, in Table 1.

Parameter name	Recommended range	Remark	Used parameters
PathLength	[1.1, 5]	Controlling parameter	3
Step	[0.11, *PathLength*]	Controlling parameter	0.11
PRT	[0, 1]	Controlling parameter	0.2
Dimemsion	Given by problem	Number of arguments in cost function	10
PopSize	[10, up to user]	Controlling parameter	10
Migrations	[10, up to user]	Stopping parameter	1000

Table 1: SOMA parameters

The run of the algorithm starts with creating an initial population. This population is randomly generated and distributed over searching space. After that, we start migrating loops. In each migrating loop, firstly we select a leader. The leader is the best individual (the individual with the best cost value). After the leader is selected, each individual jumps towards him according to (2). In this equation $x_{i,j}^{ML+1}$ is the value of i-th individual's j-th parameter in step t and migrating loop $ML + 1$. $x_{i,j,start}^{ML}$ is the value of i-th individual's j-th parameter staring position in actual migration loop. $x_{L,j}^{ML}$ is the value of leader's j-th parameter in actual migrating loop. t is a step from 0 by *Step* to *PathLength*. $PRTVector$ is a vector of ones and zeros depended on PRT. If a randomly generated number is less than PRT, then in $PRTVector$ will be 1 otherwise 0.

$$x_{i,j}^{ML+1} = x_{i,j,start}^{ML} + (x_{L,j}^{ML} - x_{i,j,start}^{ML})\, t\, PRTVector_j \tag{2}$$

Each individual remembers all positions and values of the cost function reached on these positions. After all jumps individual returns to the best position. Migrating loops repeat until we reach a maximum number of migration loops or a maximum number of evaluations.

Pseudocode of SOMA, we can see as Algorithm 1. For more information about this algorithm please look at [4, 16].

Algorithm 1 SOMAAllToOne

x: the initial randomly generated population
Controlling parameters
f_{cost}: cost function (fitness function)
evaluate initial population
migrationCycle = 0
while numberOfEvaluation < Migrations **do**
  Selection of the best individual - Leader
  while j ≤ PopSize **do**
    selection of j_{th} individual calculate f_{cost} of the new positions save the best solution of the j_{th} individual on its trajectory in a new population

4 CEC 2014 benchmark functions

For experiments we used well known CEC 2014 benchmark functions [9]. These test functions compose of 2 basic unimodal functions, 3 multimodal functions, 3 hybrid function and 7 compositional functions. We can see brief description in Table 2. For more details about the functions please look at [9].

Type	No.	Description
Unimodal functions	CF1	Rotated High Conditioned Elliptic Function
	CF2	Rotated Bent Cigar Function
Simple Multimodal functions	CF3	Shifted and Rotated Ackleys Function
	CF4	Shifted and Rotated Rastrigins Function
	CF5	Shifted and Rotated Schwefels Function
Hybrid functions	CF6	Hybrid Function 1 (N=3)
	CF7	Hybrid Function 2 (N=4)
	CF8	Hybrid Function 3 (N=5)
Composition functions	CF9	Composition Function 1 (N=3)
	CF10	Composition Function 2 (N=3)
	CF11	Composition Function 3 (N=5)
	CF12	Composition Function 4 (N=5)
	CF13	Composition Function 5 (N=5)
	CF14	Composition Function 6 (N=7)
	CF15	Composition Function 7 (N=10)

Table 2: CEC 2015 benchmark functions

5 SOMA time series

In the SOMA, all individuals travel to the leader in each migrating loop. Individuals can improve themselves during the travel. The improvements can be described relatively according to the leader. Relative improvement I can be counted according to (3), where f_I^S is the starting cost value of the jumping individual, f_I^E is the ending cost value of the jumping individual and f_L is the cost value of the leader.

$$I = \frac{f_I^S - f_I^E}{f_I^S - f_L} \tag{3}$$

If we write down the improvements of a selected individual as series, we will get the time series of improvements. Example of the time series we can see in Figure 3.

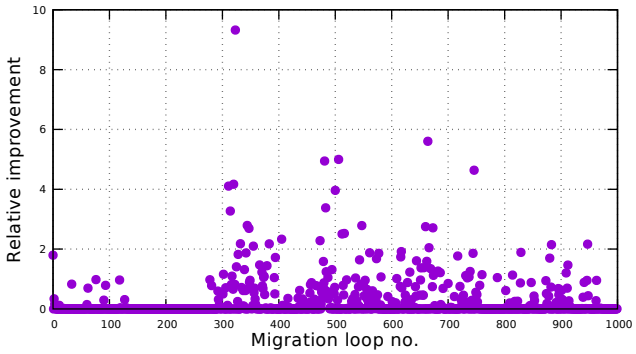


Figure 3: Time series of one individual on CF10

6 Creation of NVG from SOMA

As it has been written, we can convert the series into the NVG. Visualization of the network from time series in Figure 3 we can see in Figure 4. This graph was created in the tool called Gephi [2], and we selected ForceAtlas2 algorithm [6] as the layout. The size of vertices represents the degree. Bigger vertices have higher degree and smaller vertices have lower degree. The degree of a vertex represents the number of edges attached to it. We can count degree according to (4). k_i represents the degree k of the vertex i and $e_{i,j}$ represents the edge between vertex i and vertex j. If there is the edge between the vertex i and the vertex j, the $e_{i,j}$ equals to 1. Otherwise the $e_{i,j}$ is 0.

$$k_i = \sum_{j,i \neq j}^{|V|} e_{ij}. \tag{4}$$

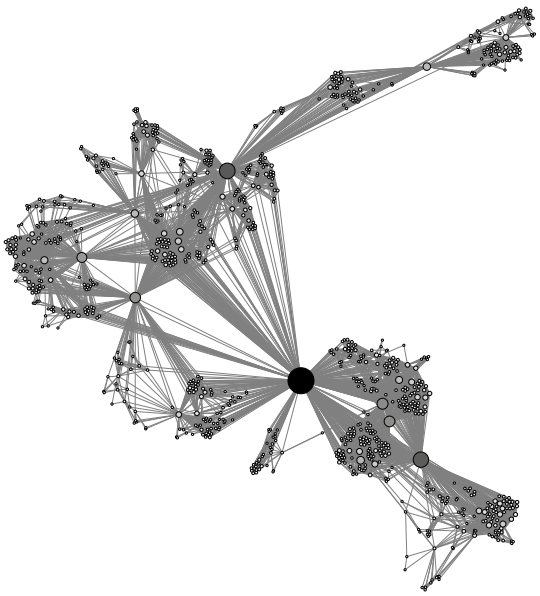


Figure 4: Network for time series from Figure 3

7 CN analysis - global network properties

Given networks can be analyzed. We can use well-developed theory about the complex networks [1, 3, 10, 14] for analyzing our time series and we can get some new information about the individuals and consequently

about the algorithm. In this section, we look at one of the most fundamental network global properties: average degree, characteristic path length, diameter, clustering coefficient.

Average degree k is given as average of all degrees in the network. We can count it according to 5, where k_i represents the degree of vertex i and $|V|$ represents the number of vertices.

$$k = \frac{1}{|V|} \sum_{i}^{|V|} k_i \tag{5}$$

Distance d_i, j is the shortest path between vertex i and vertex j. It represents the minimal number of edges we have to go through to get from vertex i to vertex j.

Characteristic path length L, also known as average path length, is given as an average of all distances between all vertices.

$$L = \frac{1}{|V| \cdot (|V| - 1)} \sum_{i}^{|V|} \sum_{j,i \neq j}^{|V|} d_{i,j} \tag{6}$$

Diameter D represents the maximal distance in the network.

Clustering coefficient CC, also known as a transitivity, represents the probability that neighbors of a vertex are also connected by the edge. We count it as a ratio between a number of triangles in graph and number triples connected by the edge.

All counted global network properties for one run and one individual on CEC 2016 benchmark functions, we can see in Table 3. As we can see, individual on all functions have similar clustering coefficient in range (0.81 - 0.92). Only for function 3, the clustering coefficient is significantly lower (0.70). Other properties differ significantly more. The average degree is in range (4.38 - 47.36), the characteristic path length is in range (2.00 - 3.20), and the diameter is in range (3 - 7). We can use these three properties for later investigation. For example, what is the optimal average degree for the best run? Is there any relationship between the final solution, network properties, and algorithm parameters? Have all individuals got same network properties or not? Is it better to have same properties higher or lower? With these answers, we can change the parameters during the run of algorithm or we can replace the bad individuals and make the algorithm batter and faster.

Type	**No.**	k	L	D	CC
Unimodal functions	CF1	10.65	2.53	6	0.85
	CF2	10.62	2.44	7	0.83
Simple Multimodal functions	CF3	4.38	2.14	4	0.70
	CF4	47.36	2.00	4	0.85
	CF5	9.09	2.23	6	0.86
Hybrid functions	CF6	12.58	2.66	5	0.84
	CF7	17.56	3.20	6	0.85
	CF8	10.21	2.64	5	0.82
Composition functions	CF9	24.54	2.02	6	0.81
	CF10	13.75	2.93	6	0.83
	CF11	9.88	2.00	3	0.92
	CF12	11.55	2.17	4	0.88
	CF13	6.76	2.11	4	0.85
	CF14	11.03	2.17	5	0.86
	CF15	13.18	2.51	6	0.84

Table 3: Global networks properties - one run and one individual

8 CN analysis - degree distribution

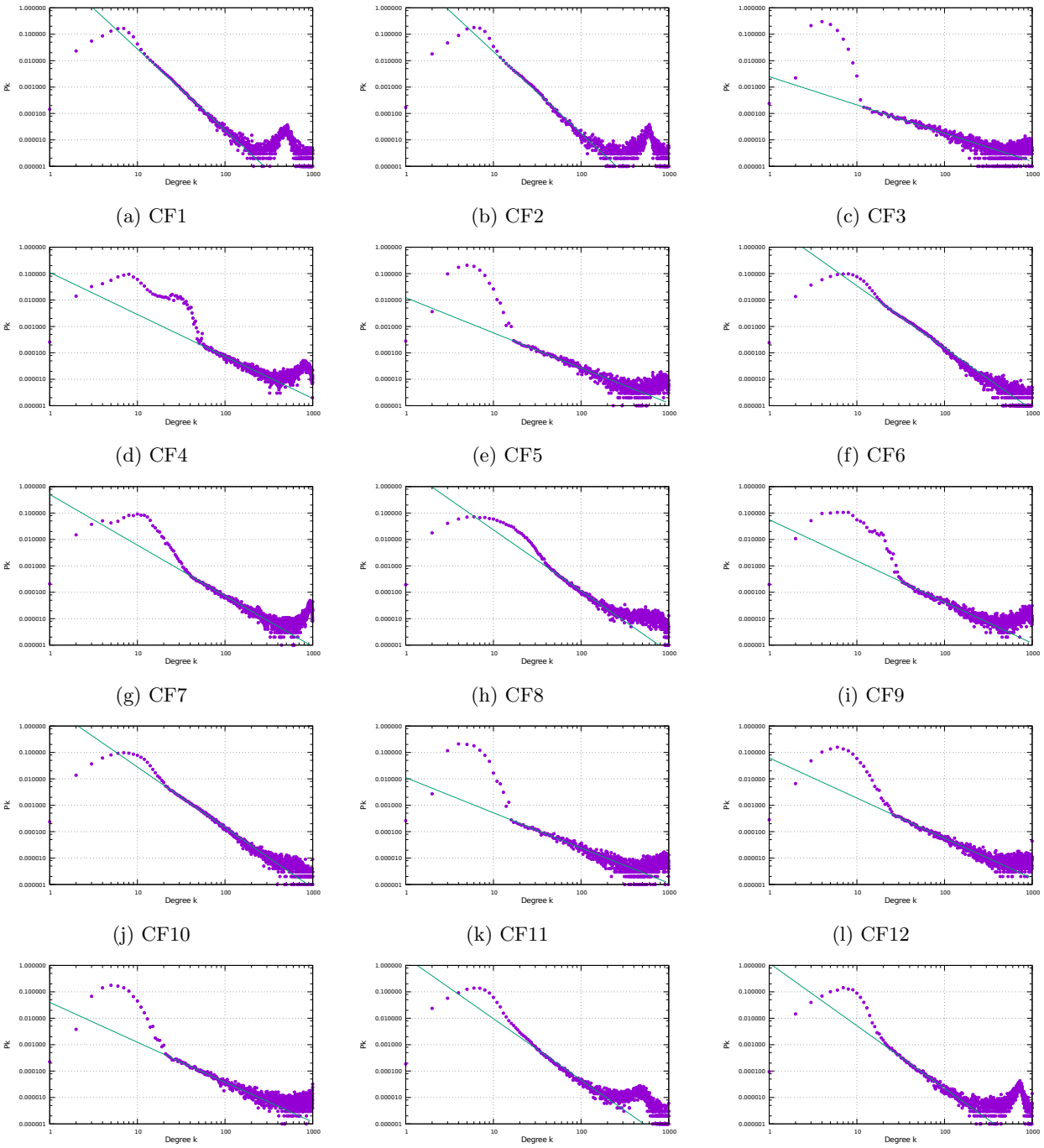
Now, we will focus on degree distribution. **Degree distribution** represents the probability distribution of degrees over the whole network. Degree was described above. Also, we can define mark p_k. p_k represents the probability that a randomly selected vertex has a degree equals to k.

In many complex networks, degree distribution meets so-called power law. It means that if we use logarithmic scales for the degree distribution, the plot will create the line. Mathematically, the degree meets the (7). k represents the degree, p_k represents the probability that a randomly selected vertex has a degree equals to k and α and c are the constants. Also, we can write the relation as (8), where C is a constant and it is equals to e^c.

$$\ln p_k = -\alpha \ln k + c \tag{7}$$

$$p_k = Ck^{-\alpha} \tag{8}$$

In Figure 5, we can see average degree distribution for 100 individuals, from 10 runs of SOMA. Also, we can see a line in the figure, which denotes the power law. Power law can be observed on each function for a small interval. Generally, each degree distribution function can be divided into three parts. The middle part meets the power law, and the other two parts do not fit into this line. In these parts, the probabilities are higher.



(a) CF1 (b) CF2 (c) CF3

(d) CF4 (e) CF5 (f) CF6

(g) CF7 (h) CF8 (i) CF9

(j) CF10 (k) CF11 (l) CF12

(m) CF13 (n) CF14 (o) CF15

Figure 5: Average degree distribution for 100 individuals from 10 runs

The lines from the Figure 5 can be described with the constants α and c according to the Equation (7). You can find these constants in Table 4. Also, there is a column with degree k, which represents the lowest degree, which belongs to the power law.

Type	No.	α	c	k
Unimodal functions	CF1	3.07	3.5	12
	CF2	3.09	3.3	12
Simple Multimodal functions	CF3	1.07	-6.0	12
	CF4	1.59	-2.2	48
	CF5	1.33	-4.4	17
Hybrid functions	CF6	2.35	2.1	18
	CF7	1.92	-0.7	34
	CF8	2.31	1.6	36
Composition functions	CF9	1.60	-2.9	32
	CF10	2.29	1.7	21
	CF11	1.33	-4.5	16
	CF12	1.52	-2.8	26
	CF13	1.51	-3.2	21
	CF14	2.32	0.7	30
	CF15	2.37	0.2	22

Table 4: Power law constants

The shape of the degree distribution function raises additional questions. What are the optimal coefficients of the power law? Is there any relationship between the tail of the curve and the cost function?

9 Conclusion

In our research, we are attempting to convert run of evolution algorithm into a network. In previous articles, we mainly focused on interactions networks. In these networks, we captured interactions between individuals in the population. Interactions like crossing or leader-individual interactions. In this article, we looked at the different type of networks. We captured the individual's relative improvements into time series and then we converted these series into the NVG. We mainly focused on SOMA.

We showed how the time series and network may look like on a selected function. Also, we counted basic network properties and propose how this results can be used for next investigations.

This article showed us a new view on evolution algorithms. It presented a new possibility how can we analyze the evolution algorithms. We could observe different networks for different cost functions. For the later investigation, we can focus on the correlation between the networks and the performance of the evolution algorithm. After we can use a different type of SOMA on a different problem or we can use different parameters or a different algorithm. According to networks, we can switch between algorithms, parameters, and versions (strategies) of the algorithms.

Also, many optimization problems are solved as a black box. We can use SOMA for classification of this problem. As we saw, for different types of problems, we get different properties and different degree distribution. So with this analysis and comparison of well-known problems, we can tell something about the black box problem. Is the black box problem unimodal or multimodal function? How complex is this problem?

Acknowledgement: The following grants are acknowledged for the financial support provided for this research: Grant Agency of the Czech Republic - GACR P103/15/06700S, Grant of SGS No. SGS 2017/134, VSB-Technical University of Ostrava.

References

[1] Albert, R., Barabási, A.L.: Statistical mechanics of complex networks. Reviews of modern physics **74**(1), 47 (2002)

[2] Bastian, M., Heymann, S., Jacomy, M., et al.: Gephi: an open source software for exploring and manipulating networks. ICWSM **8**, 361–362 (2009)

[3] Boccaletti, S., Latora, V., Moreno, Y., Chavez, M., Hwang, D.U.: Complex networks: Structure and dynamics. Physics reports **424**(4), 175–308 (2006)

[4] Davendra, D., Zelinka, I., et al.: Self-organizing migrating algorithm. New Optimization Techniques in Engineering (2016)

[5] Holland, J.H.: Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT press (1992)

[6] Jacomy, M., Venturini, T., Heymann, S., Bastian, M.: Forceatlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software. PloS one **9**(6), e98,679 (2014)
[7] Janostik, J., Pluhacek, M., Senkerik, R., Zelinka, I., Spacek, F.: Capturing inner dynamics of firefly algorithm in complex networkinitial study. In: Proceedings of the Second International Afro-European Conference for Industrial Advancement AECIA 2015, pp. 571–577. Springer (2016)
[8] Lacasa, L., Luque, B., Ballesteros, F., Luque, J., Nuno, J.C.: From time series to complex networks: The visibility graph. Proceedings of the National Academy of Sciences **105**(13), 4972–4975 (2008)
[9] Liang, J., Qu, B., Suganthan, P., Chen, Q.: Problem definitions and evaluation criteria for the cec 2015 competition on learning-based real-parameter single objective optimization. Technical Report201411A, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore (2014)
[10] Newman, M.E.: The structure and function of complex networks. SIAM review **45**(2), 167–256 (2003)
[11] Skanderova, L., Fabian, T.: Differential evolution dynamics analysis by complex networks. Soft Computing pp. 1–15 (2015)
[12] Skanderova, L., Zelinka, I.: Differential evolution dynamic analysis in the form of complex networks. In: Advanced Methods for Complex Network Analysis, pp. 285–318. IGI Global (2016)
[13] Storn, R., Price, K.: Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization **11**(4), 341–359 (1997)
[14] Strogatz, S.H.: Exploring complex networks. Nature **410**(6825), 268–276 (2001)
[15] Tomaszek, L., Zelinka, I.: On performance improvement of the soma swarm based algorithm and its complex network duality. In: Evolutionary Computation (CEC), 2016 IEEE Congress on, pp. 4494–4500. IEEE (2016)
[16] Zelinka, I.: Somaself-organizing migrating algorithm. In: New optimization techniques in engineering, pp. 167–217. Springer (2004)
[17] Zelinka, I.: On mutual relations amongst evolutionary algorithm dynamics and its hidden complex network structures: An overview and recent advances. Nature-Inspired Computing: Concepts, Methodologies, Tools, and Applications: Concepts, Methodologies, Tools, and Applications p. 215 (2016)
[18] Zelinka, I., Davendra, D.D., Chadli, M., Senkerik, R., Dao, T.T., Skanderová, L.: Evolutionary dynamics as the structure of complex networks. In: Handbook of Optimization, pp. 215–243. Springer (2013)
[19] Zelinka, I., Tomaszek, L., Kojecky, L.: On evolutionary dynamics modeled by ant algorithm. In: Intelligent Networking and Collaborative Systems (INCoS), 2016 International Conference on, pp. 193–198. IEEE (2016)