Lenka Skanderova*, Tomas Fabian and Ivan Zelinka

Differential Evolution Dynamics Modeled by Longitudinal Social Network

DOI 10.1515/jisys-2015-0140

Received October 29, 2015; previously published online June 21, 2016.

Abstract: Differential evolution (DE) is a population-based algorithm using Darwinian and Mendel principles to find out an optimal solution to difficult problems. In this work, the dynamics of the DE algorithm are modeled by using a longitudinal social network. Because a population of the DE algorithm is improved in generations, each generation of DE algorithm is represented by one short-interval network. Each short-interval network is created by individuals contributing to population improvement. On the basis of this model, a new parent selection in the mutation operation is presented and a well-known benchmark set CEC 2013 Special Session on Real-Parameter Optimization (including 28 functions) is used to evaluate the performance of the proposed algorithm.

Keywords: Differential evolution dynamics, longitudinal social network, degree centrality.

2010 Mathematics Subject Classification: 68T99.

1 Introduction

Differential evolution (DE) is an efficient algorithm for optimization introduced by Price and Storn in 1997 [10]. DE is easy to use, and its convergence properties are considered to be very good. Despite these facts, DE also has several disadvantages: its convergence is unstable and it easily drops into the local extreme [14]. To eliminate or even remove these bottlenecks is the goal of many researchers from all over the world.

Brest et al. described DE using self-adaptive control parameters (jDE), where each individual is extended with its own control parameters F_i (scale parameter) and CR_i (crossover rate) [2]. Wang et al. proposed the DE with composite trial vector generation strategies and control parameters [13], where three mutation strategies are combined with three different control parameter settings to create three trial vectors. Zhou et al. introduced the DE with an intersect mutation operator called IMDE [17]. In this algorithm, the populations of individuals are divided into better and worse parts according to their objective function value. Zhou et al. also presented a new DE with a hybrid mutation operator and self-adapting control parameters for global optimization (HSDE), where each individual is enhanced by its own scale parameter F_i and crossover rate CR_i as in the case of jDE. Then, for each target vector, a mutation vector is created such that five mutually different parents are selected randomly. If an objective function value of a target vector is better than an objective function value of the first two parents, the DE/rand/1/bin strategy is applied to generate a mutation vector; otherwise, DE/current-to-best/1/bin is applied.

Most of above-mentioned researches deal with three ways on how to improve DE. The first one is to propose more efficient mutation operators, the second one is focused on the control parameter settings, and

*Corresponding author: Lenka Skanderova, Department of Computer Science, Faculty of Electrical Engineering and Computer Science, VSB, Technical University of Ostrava, 17. listopadu 15/2172, 70833 Ostrava, Czech Republic, e-mail: lenka.skanderova@vsb.cz

Tomas Fabian and Ivan Zelinka: Department of Computer Science, Faculty of Electrical Engineering and Computer Science, VSB, Technical University of Ostrava, 17. listopadu 15/2172, 70833 Ostrava, Czech Republic

the third method deals with a process of parent selection in a mutation operation. In this work, a longitudinal social network (LSN) is used to model the DE dynamics. The main motivation is to better understand relationships between individuals and use of this knowledge to improve the DE performance.

The idea to model evolutionary algorithm dynamics using a complex network (CN) was presented by Zelinka et al. in 2010, where the authors investigated a possibility to visualize the dynamics of evolutionary algorithms by using CNs [16]. One year later, Zelinka et al. visualized evolutionary algorithm dynamics by using CNs; moreover, the authors visualized the dynamics of the created CNs using the coupled map lattices [15]. In 2014, Davendra et al. analyzed the CNs generated by Discrete Self-Organizing Migrating Algorithm, where a permutation flow-shop scheduling problem with blocking constraint had been used as the test problem [4].

The last mentioned publications [4, 15, 16] dealt with possibility to create CNs by evolutionary algorithm dynamics and with the analysis of such generated CNs. However, we have found no research where the dynamics of an evolutionary algorithm is modeled by LSN, and the results of this analysis are used to improve the evolutionary algorithm's performance. In this work, an LSN is used to model the dynamics of the DE algorithm. An LSN consists of the snapshots of the network capturing the state of the network in different points of time. These snapshots are called short-interval networks (SINs). In this work, each generation is modeled by one SIN to capture positive relationships between individuals in a population established in this generation. On the basis of this model, a new parent selection in the mutation operation is presented, where the parents are selected on the basis of the strength of nodes representing these individuals in the LSN.

The rest of the paper is organized as follows. The DE algorithm is described in Section 1.1. In Section 2, modeling of the DE dynamics by LSN is clarified and the novel principle of parent selection in the mutation operation is described. The results provided by the original version as well as by the novel DE algorithm are provided in Sections 3 and 4.

1.1 Differential Evolution

DE works with a population of NP individuals $\bar{x}_i^G$, $i = \{1, \dots, NP\}$. G denotes a generation. One individual $\bar{x}_i^G$ consists of D parameters. Each parameter is constrained by its search range $[x_{\text{lower},j}, x_{\text{upper},j}]$, where lower and upper denote lower and upper bounds for each parameter and $j = \{1, \dots, D\}$ is an index of a parameter. The first population is generated randomly in the space of possible solutions. Then, for each target vector $\bar{x}_i^G$, three mutually different solution vectors (parents), $\bar{x}_{r_1}^G$, $\bar{x}_{r_2}^G$, and $\bar{x}_{r_3}^G$, are selected randomly to create a mutation vector $\bar{v}_i^G$ [see Eq. (1)]. In the crossover operation, parameters of a mutation vector are combined with parameters of a target vector on the basis of the crossover rate CR to create a trial vector $\bar{u}_i^G$ [see Eq. (2)]. If an objective function value of a trial vector $f(\bar{u}_i^G)$ is better than an objective function value of a target vector $f(\bar{x}_i^G)$, a trial vector (offspring) will survive to the next generation; otherwise, a target vector will survive [see Eq. (3)].

The mutation operation is mathematically defined by the following equation:

$$\bar{v}_i^G = \bar{x}_{r_1}^G + F \cdot (\bar{x}_{r_2}^G - \bar{x}_{r_3}^G), \quad (1)$$

where F is the scale parameter with a typical value between 0.4 and 0.95 [7].

The crossover operation is then described by the following equation:

$$\bar{u}_{i,j}^G = \begin{cases} \bar{v}_{i,j}^G & \text{if } r(j) \leq CR \text{ or } j = m(i) \\ \bar{x}_{i,j}^G & \text{if } r(j) > CR \text{ and } j \neq m(i), \end{cases} \quad (2)$$

where $CR \in [0, 1]$, $m(i) \in \{1, \dots, D\}$ is an integer selected randomly with the uniform distribution ensuring that a trial vector $\bar{u}_i^G$ will contain at least one parameter from a mutation vector $\bar{v}_i^G$, and $r_j \in [0, 1]$ represents a random number from the unit interval generated for each j -th parameter [17].

$$\bar{x}_i^{G+1} = \begin{cases} \bar{u}_i^G & \text{if } f(\bar{u}_i^G) \leq f(\bar{x}_i^G) \text{ (for minimization problem)} \\ \bar{x}_i^G & \text{otherwise.} \end{cases} \quad (3)$$

2 LSN Created by DE Dynamics

In Refs. [15, 16], Zelinka et al. described the principle of CN creation by selected evolutionary algorithm dynamics. This principle accepts the philosophy that individuals of a population move from worse to better positions (similarly to individuals in swarm algorithms), which means that theoretically, a population consists of the same individuals just changing their positions. More precisely, a new offspring creation is considered to be just an activation of a target vector $\vec{x}_i^G$ to move to a better position $\vec{x}_i^{G+1}$.

As it was mentioned in Section 1, in this work, LSNs are used to model the dynamics of the DE algorithm. LSN consists of a number of different snapshots, so called SINs, capturing a state of the social network at different points in time. LSNs have been successfully used in different areas of research to better understand the difficult processes in various contexts [12]; for example, smoking behavior among adolescents in British schools [8], dynamic spread of happiness in a large social network [5], the spread of obesity in a large social network [3], virus spread in computer networks [9], etc.

In this work, each generation of the DE algorithm is modeled by one SIN. Each SIN captures only positive relationships between individuals in a population. In the case that a mutation vector $\vec{v}_i^G$ (generated by individuals $\vec{x}_{r_1}^G$, $\vec{x}_{r_2}^G$, and $\vec{x}_{r_3}^G$) and a target vector $\vec{x}_i^G$ generate a trial vector with better objective function value than an objective function value of a target vector, a relationship of a target vector $\vec{x}_i^G$ to solution vectors $\vec{x}_{r_1}^G$, $\vec{x}_{r_2}^G$, and $\vec{x}_{r_3}^G$ will be considered to be positive, and a target vector $\vec{x}_i^G$ will be modeled as a source while solution vectors $\vec{x}_{r_1}^G$, $\vec{x}_{r_2}^G$, and $\vec{x}_{r_3}^G$ will be modeled as destinations of an arc. Any other type of relationship between individuals will not be reflected.

2.1 Parent Selection on the Basis of Node Strength

In the previous section, an LSN creation on the basis of DE dynamics has been described. As mentioned, for each generation, one SIN is created. Thanks to this mechanism, positive relationships between individuals established in the generation are captured. Each SIN is created by the nodes represented only by individuals contributing to improvement of a population. This means that the number of individuals in two consecutive SINs can be different. The number of arcs leading to the node representing a solution vector $\vec{x}_k^G$ can be interpreted as the number of new offspring created, thanks to this individual. The number of arcs leading from the node representing a target individual $\vec{x}_i^G$ can then be interpreted as the number of individuals creating a mutation vector $\vec{v}_i^G$. Because an oriented network is created, from the social network analysis point of view, the in-degree and out-degree of nodes are discussed in the following text.

Generally, a degree of the node i is the number of edges connecting this node with the other nodes, which is defined by the following equation:

$$k_i = \sum_{j \in N} a_{ij}, \quad (4)$$

where a_{ij} is the element of an adjacency matrix defined as 1 if a node i is connected to a node j , and 0 otherwise. In oriented graphs, in-degree is then defined as $k_i^{in} = \sum_j a_{ji}$ and out-degree as $k_i^{out} = \sum_j a_{ij}$. The total degree of a node i is defined as $k_i = k_i^{out} + k_i^{in}$ [1]. The degree centrality of a node is its degree. To compare the degree centrality of different nodes in networks with different number of nodes, the in-degree and out-degree centrality is given by Eq. (5), where N is the total number of nodes in a network [11].

$$\begin{aligned} k_i^{in} &= \frac{\sum_j a_{ji}}{N-1} \\ k_i^{out} &= \frac{\sum_j a_{ij}}{N-1}. \end{aligned} \quad (5)$$

For the weighted networks, the node strength s_i is defined as

$$s_i = \sum_j^N w_{ij}, \quad (6)$$

where w_{ij} is an element of a weighted adjacency matrix and $w_{ij} > 0$ if a node i is connected to a node j . The value of w_{ij} represents the weight of the edge. For weighted oriented networks, in-strength s_i^{in} and out-strength s_i^{out} are defined similarly to in-degree and out-degree in oriented networks.

In 2011, Uddin et al. introduced a time scale degree centrality, taking into consideration the presence as well as duration of links among actors within a network. The work of Uddin et al. from 2015 [12] proposed three new measures to quantify the dynamicity of LSNs: positional dynamicity, participation dynamicity, and complete dynamicity. Positional dynamicity represents changes of actor positions in different SINs relative to positions of these actors in the aggregated network. Participation dynamicity illustrates the actors changing network participation in any two consecutive SINs. Finally, the complete dynamicity of an actor is given by the summation of its positional and participation dynamicity. These measures seem to be very promising in the future research of the LSN generated by evolutionary algorithm dynamics.

From Eq. (1), it is obvious that the out-degree of each node in SIN will be 0 or 3. On the other hand, the in-degree of each node will be different because one solution vector can be used to create a mutation vector more than once. A node with the highest in-degree in the SIN representing a generation G will represent the solution vector, which has been used to create a new offspring in the generation G most often. On the other hand, the out-degree 3 of a node means that the individual represented by this node has been replaced by better offspring and its properties might positively influence a population evolution in the next generation.

In the following experiment, the mutation operator has been modified such that one solution vector can be selected simultaneously as the solution vectors $\bar{x}_{r_1}^G$, $\bar{x}_{r_2}^G$, and $\bar{x}_{r_3}^G$ in the mutation operation. This means that a mutation vector can be generated by three mutually different solution vectors [see Eq. (1)]; however, it can be also generated by only two mutually different solution vectors [see Eqs. (7) and (8)]. Finally, only one solution vector can be selected to be a mutation vector [see Eq. (9)]. In such case, there is no mutation; the parameters of a target vector are just combined with the parameters of this solution vector.

$$\begin{aligned} - \quad & \bar{x}_{r_1}^G = \bar{x}_{r_2}^G \text{ and } \bar{x}_{r_1}^G \neq \bar{x}_{r_3}^G \\ & \bar{v}_i^G = \bar{x}_{r_1}^G + F(\bar{x}_{r_1}^G - \bar{x}_{r_3}^G), \end{aligned} \quad (7)$$

$$\begin{aligned} - \quad & \bar{x}_{r_1}^G = \bar{x}_{r_3}^G \text{ and } \bar{x}_{r_1}^G \neq \bar{x}_{r_2}^G \\ & \bar{v}_i^G = \bar{x}_{r_1}^G + F(\bar{x}_{r_1}^G - \bar{x}_{r_2}^G), \end{aligned} \quad (8)$$

$$\begin{aligned} - \quad & \bar{x}_{r_1}^G \neq \bar{x}_{r_3}^G \text{ and } \bar{x}_{r_1}^G \neq \bar{x}_{r_2}^G \text{ and } \bar{x}_{r_2}^G = \bar{x}_{r_3}^G \\ & \bar{v}_i^G = \bar{x}_{r_1}^G. \end{aligned} \quad (9)$$

In the case of Eqs. (7) and (8), two arcs would lead from the node representing a target vector $\bar{x}_i^G$ to a node representing a solution vector $\bar{x}_{r_1}^G$. In the case of Eq. (9), even three arcs would lead from the node representing a target vector $\bar{x}_i^G$ to the node representing a solution vector $\bar{x}_{r_1}^G$. These multiple arcs are replaced by one weighted arc. This means that in the case of Eqs. (7) and (8), the weight of the arc leading from the node representing a target vector $\bar{x}_i^G$ to the node representing a solution vector $\bar{x}_{r_1}^G$ will be two. Moreover, in the case of Eq. (9), the weight of the arc leading from the node representing a target vector $\bar{x}_i^G$ to the node representing a solution vector $\bar{x}_{r_1}^G$ will be three. At the end of each generation, each individual will remember the strength of the node, which represents this individual in the LSN, and in the next generation, for each target vector, three solution vectors are selected on the basis of the node strength. A higher node strength means a higher probability to be selected for the mutation operation.

3 Experimental Results

The algorithm is implemented in C#, Microsoft.NET Framework 4.5.1, and run on a computer with CPU Intel(R) Xeon(R) 2.83 GHz. To evaluate the performance of the presented algorithm, 28 benchmark functions from the CEC 2013 [6] have been used.

In our work, the number of problem dimensions is set to 30 for all 28 test functions. The detailed parameter settings are given as follows. The population size has been set to 100 for each benchmark function, and 50 independent runs have been conducted with 300,000 function evaluations as the termination criterion. The scale parameter has been set to $F = 0.6$ and the crossover rate to $CR = 0.9$. The comparison of the original version of DE (denoted as DE/rand/1/bin) and the new version of DE (denoted as DE/deg/1/bin) described in detail in the previous section and the related experimental results of benchmark set are shown in Table 1, where “Mean Error” and “Std Dev” indicate the average and standard deviation of the function error values obtained in 50 independent runs. To make a fair comparison, the Wilcoxon signed-rank test at the $\alpha = 0.05$ level is conducted between DE/rand/1/bin and DE/deg/1/bin.

DE/deg/1/bin provided better results in all five unimodal functions f_1 – f_5 (see Table 1). Better results have also been achieved for the step function f_6 , noisy quartic function f_7 , and multimodal functions f_9 , and f_{11} – f_{20} ,

Table 1: Experimental Results of DE/rand/1/bin and DE/deg/1/bin over 50 Independent Runs on 28 Test Functions of 30 Variables with 300,000 Function Evaluations.

Function	DE/rand/1/bin		DE/deg/1/bin		
	Mean Error	Std Dev	Mean Error	Std Dev	
Unimodal functions					
f_1	3.80E−12	5.63E−12	1.01E−29	2.13E−29	+
f_2	1.59E+06	6.74E+05	4.66E+05	3.26E+05	+
f_3	5.29E+04	4.14E+04	3.41E+03	2.34E+04	+
f_4	1.73E+04	4.36E+03	2.01E+03	8.89E+02	+
f_5	3.16E−08	1.83E−08	4.33E−25	8.99E−25	+
Multimodal functions					
f_6	9.27E+00	2.67E+00	6.24E+00	3.33E+00	+
f_7	6.92E+00	2.51E+00	1.78E−01	2.25E−01	+
f_8	2.09E+01	5.20E−02	2.10E+01	6.19E−02	≈
f_9	3.94E+01	9.38E−01	1.20E+01	4.21E+00	+
f_{10}	2.17E−03	2.59E−03	1.01E−02	5.33E−03	−
f_{11}	1.77E+02	1.60E+01	1.85E+01	5.52E+00	+
f_{12}	1.99E+02	1.08E+01	3.63E+01	1.67E+01	+
f_{13}	2.00E+02	1.26E+01	5.88E+01	2.70E+01	+
f_{14}	6.65E+03	4.27E+02	7.71E+02	2.58E+02	+
f_{15}	7.19E+03	3.00E+02	5.92E+03	1.08E+03	+
f_{16}	2.47E+00	2.47E−01	2.36E+00	3.54E−01	+
f_{17}	2.06E+02	1.89E+01	5.60E+01	5.62E+00	+
f_{18}	2.06E+02	1.88E+01	5.11E+01	5.75E+00	+
f_{19}	1.68E+01	8.98E−01	3.05E+00	6.95E−01	+
f_{20}	1.25E+01	2.42E−01	1.19E+01	4.79E−01	+
Composition functions					
f_{21}	3.39E+02	4.90E+01	3.37E+02	4.85E+01	+
f_{22}	6.96E+03	3.87E+02	7.27E+02	2.23E+02	+
f_{23}	7.62E+03	2.84E+02	6.03E+03	9.93E+02	+
f_{24}	2.04E+02	1.07E+00	2.00E+02	1.56E−01	+
f_{25}	1.97E+02	2.83E+01	2.31E+02	2.07E+01	+
f_{26}	2.00E+02	2.52E−02	2.00E+02	9.52E−03	+
f_{27}	4.40E+02	9.10E+01	3.05E+02	5.89E+00	+
f_{28}	3.00E+02	5.39E−05	3.00E+02	0.00E+00	+

Notations “+,” “≈,” and “–” indicate that DE/deg/1/bin provided better, similar, or worse results than DE/rand/1/bin.

and for all composition functions $f_{21}-f_{28}$. In the case of the multimodal test function f_8 , the results are comparable. The DE/deg/1/bin provided worse results only in one from 28 test functions from the benchmark set, namely in the case of the multimodal function f_{10} .

4 Conclusion

In this work, the LSN is used to model the dynamics of the DE algorithm. Each SIN is created only by individuals contributing to the improvement of the population and represents one generation of the DE algorithm. On the basis of this model, a new principle of the solution vector selection to generate a mutation vector has been presented. For each generation, one SIN is created and the strength of each node is calculated. In the next generation, the solution vectors are selected on the basis of this strength, which means that individuals with a higher node strength have a higher probability to be selected in the mutation operation to create a mutation vector. In this novel method, a mutation vector can be generated by two or three different solution vectors, or the mutation vector is replaced by one randomly selected solution vector. In this case, parameters of a target vector are just combined with parameters of this one solution vector. The performance of this novel algorithm has been tested on a 28-test function from the well-known benchmark CEC 2013, and the results have been compared with the results provided by the original version of DE. The novel algorithm denoted as DE/deg/1/bin provided better results in most of the test functions, more precisely in 26 from 28 test functions. In one case, the results were comparable and in one case the novel algorithm has provided worse results than the original version of DE.

Acknowledgments: The following grants are acknowledged for the financial support provided for this research: Grant Agency of the Czech Republic – GACR P103/15/06700S, partially supported by Grant of SGS No. SP2015/142, VSB, Technical University of Ostrava.

Bibliography

- [1] S. Boccaletti, V. Latorab, Y. Morenod, M. Chavezf and D. U. Hwanga, Complex networks: structure and dynamics, *Phys. Rep.* **424** (2006), 175–308.
- [2] J. Brest, S. Greiner, B. Boskovic, M. Mernik and V. Zumer, Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems, *IEEE Trans. Evolut. Comput.* **10** (2006), 646–657.
- [3] N. A. Christakis and J. H. Fowler, The spread of obesity in a large social network over 32 years, *N. Eng. J. Med.* **357** (2007), 370–379.
- [4] D. Davendra, I. Zelinka, R. Senkerik and M. Pluhacek, Complex network analysis of discrete self-organising migrating algorithm, in: *Nostradamus 2014: Prediction, Modeling and Analysis of Complex Systems*, pp. 161–174, 2014.
- [5] J. H. Fowler and N. A. Christakis, Dynamic spread of happiness in a large social network: longitudinal analysis over 20 years in the Framingham Heart Study, *BMJ* **337** (2008), a2338.
- [6] J. J. Liang, B. Y. Qu, P. N. Suganthan and A. G. Hernández-Daz, Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization, *Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical Report 201212*, 2013.
- [7] R. Mallipeddi, P. N. Suganthan, Q. K. Pan and M. F. Tasgetiren, Differential evolution algorithm with ensemble of parameters and mutation strategies, *Appl. Soft Comput.* **11** (2011), 1679–1696.
- [8] L. Mercken, C. Steglich, P. Sinclair, J. Holliday and L. Moore, A longitudinal social network analysis of peer influence, peer selection, and smoking behavior among adolescents in British schools, *Health Psychol.* **31** (2012), 450.
- [9] J. Ren, X. Yang, Q. Zhu, L. X. Yang and C. Zhang, A novel computer virus model and its dynamics, *Nonlinear Anal. Real World Appl.* **13** (2012), 376–384.
- [10] R. Storn and K. Price, Differential evolution – a simple and efficient adaptive scheme for global optimization over continuous spaces, in: *ICSI, IEEE*, 3 1995.
- [11] S. Uddin, L. Hossain and R. T. Wigand, New direction in degree centrality measure: towards a time-variant approach, *Int. J. Inf. Technol. Decis. Mak.* **13** (2014), 865–878.
- [12] S. Uddin, A. Khan and M. Piraveenan, A set of measures to quantify the dynamicity of longitudinal social networks, *Complexity* (2015).

- [13] Y. Wang, Z. Cai and Q. Zhang, Differential evolution with composite trial vector generation strategies and control parameters, *Evol. Comput.* **15** (2011), 55–66.
- [14] Y. C. Wu, W. P. Lee and C. W. Chien, Modified the performance of differential evolution algorithm with dual evolution strategy, in: *International Conference on Machine Learning and Computing*, 3, pp. 57–63, 2011.
- [15] I. Zelinka, D. D. Davendra, R. Senkerik and R. Jasek, Do evolutionary algorithm dynamics create complex network structures?, in: *Proceedings of the 12th WSEAS international conference on Communications*, 2011.
- [16] I. Zelinka, D. D. Davendra, V. Snasel, R. Senkerik and R. Jasek, Preliminary investigation on relations between complex networks and evolutionary algorithms dynamics, in: *Computer Information Systems and Industrial Management Applications (CISIM)*, 2010.
- [17] Y. Zhou, X. Li and L. Gao, A differential evolution algorithm with intersect mutation operator, *Appl. Soft Comput.* **13** (2013), 390–401.