An improved genetic algorithm with dynamic topology*

Kai-Quan Cai(蔡开泉)[1], Yan-Wu Tang(唐焱武)[1], Xue-Jun Zhang(张学军)[1,†], and Xiang-Min Guan(管祥民)[2]

[1] School of Electronic and Information Engineering, Beihang University, Beijing 100191, China
[2] Department of General Aviation, Civil Aviation Management Institute of China, Beijing 100102, China

(Received 10 July 2016; revised manuscript received 10 September 2016; published online 25 October 2016)

The genetic algorithm (GA) is a nature-inspired evolutionary algorithm to find optima in search space via the interaction of individuals. Recently, researchers demonstrated that the interaction topology plays an important role in information exchange among individuals of evolutionary algorithm. In this paper, we investigate the effect of different network topologies adopted to represent the interaction structures. It is found that GA with a high-density topology ends up more likely with an unsatisfactory solution, contrarily, a low-density topology can impede convergence. Consequently, we propose an improved GA with dynamic topology, named DT-GA, in which the topology structure varies dynamically along with the fitness evolution. Several experiments executed with 15 well-known test functions have illustrated that DT-GA outperforms other test GAs for making a balance of convergence speed and optimum quality. Our work may have implications in the combination of complex networks and computational intelligence.

Keywords: complex networks, genetic algorithm dynamic topology

PACS: 89.75.Hc, 05.45.–a **DOI:** 10.1088/1674-1056/25/12/128904

1. Introduction

Optimization problems arise in almost every field of science, engineering and business. Many of these problems are large-scale and nonlinear that cannot be solved analytically by traditional mathematic programming. Consequently, the evolutionary algorithm was proposed to address these problems. The evolutionary algorithm derived from biological entities and natural processes is used to search the optimum of a problem under certain conditions via the evolution of individuals. Many efforts have been dedicated to the algorithm in recent decades.[1–3]

The genetic algorithm (GA), a classical evolutionary algorithm, was proposed in 1975.[4] In GA, a population of candidate solutions to an optimization problem is evolved toward better solutions using operators inspired by the natural evolution, such as inheritance, selection, crossover, and mutation. In essence, the optimization process of the algorithm is based on information exchange among individuals. As GA has been successfully applied in a wide range of fields because of briefness, universality, and strong robustness,[5–8] lots of related work springs up.

A series of work investigated the code scheme to improve the efficiency. Binary code was adopted in the traditional GA proposed by Hollstien *et al.*[9,10] Goldberg *et al.* introduced a real-coded genetic algorithm (RC-GA) using the floating-point or other high-cardinality coding in their chromosomes.[11] Another series of work were aimed at parameter control. Srinivas *et al.* recommended an adaptive genetic algorithm (AGA),[12] wherein, the probabilities of crossover and mutation are varied depending on the fitness values of the solutions. Additionally, there were other researchers devoted to solving the multi-objective problem. In 2002, Deb *et al.* introduced a fast and elitist multi-objective genetic algorithm (NSGA-II), using a fast nondominated sorting approach, which can solve the multi-objective problems more efficiently.[13] In 2007, a multi-objective evolutionary algorithm based on decomposition (MOEA/D) was presented by Zhang.[14] Until now, a variety of latest theoretical fruits have never stopped being introduced to improving GA. From the above, most of the previous works have made a contribution to the improvement of GA through the modifications of code scheme, parameter control, and objective expression. However, to the best of our knowledge, the interaction structure among individuals has not been taken into consideration.

Much evidence has shown that the structural properties are the key parts of dynamical processes in complex networks.[15,16] In recent studies, complex networks[17–19] were employed to represent the interaction structure among individuals in evolutionary algorithms (EA). In particular, it has been discovered that the network structural properties play key roles in Particle Swarm Optimization (PSO) algorithm.[20–23] Hence, it reminds us to investigate the effects of various topologies on the performance of GA. The topologies from ring to fully-connected are used to test the performance varia-

*Project supported by the National Natural Science Foundation for Young Scientists of China (Grant No. 61401011), the National Key Technologies R & D Program of China (Grant No. 2015BAG15B01), and the National Natural Science Foundation of China (Grant No. U1533119).

†Corresponding author. E-mail: zhxjbh@163.com

© 2016 Chinese Physical Society and IOP Publishing Ltd

tion of regular topologies with different network densities. We found that GA with a high-density topology converges more likely with an unsatisfactory solution. However, GA with a low-density topology possesses a slower convergence speed. Since the high-density and low-density topology have their disadvantage respectively, the GA with a new dynamic topology, named DT-GA, is proposed in which the network density of topology varies depending on the fitness values of the solution. To demonstrate the power of the method, the algorithm is evaluated using 15 famous benchmark functions. It is found that DT-GA gives rise to a better balance between the convergence speed and the optimum quality, outperforming the traditional GA.

The rest of the paper is structured as follows: Section 2 reports the effects of various topologies on the performance of GA and provides the method of GA with dynamic topology. Section 3 presents the performance of DT-GA and other test GAs. Section 4 gives the conclusion.

2. Materials and methods

2.1. Genetic algorithm

GA is one of the evolutionary algorithms that mimics the process of natural evolution and selection. Theoretically speaking, GA is a dynamical iterative process. During the iterative process, GA proceeds to initialize a population of solutions and then to improve it through recurrent application of genetic operators including selection, crossover, and mutation operator. Commonly, the algorithm terminates when either a maximum number of generations has been reached, or a satisfactory fitness level has been reached for the population.

Selection is one of the important operators existing in GA. In each generation, the fitness value of every individual in the population is evaluated. The individuals with high fitness value are stochastically selected from the current population and then form a new generation. Mutation represents the changes in gene of an individual. The role of mutation is to provide a guarantee that the search algorithm is not trapped at a local optimum.[24]

Crossover operator inevitably involves generating new individuals for improving search efficiency in the solution space. Crossover operator is introduced for two individuals (parents), so that they can combine to produce two more individuals (children). In crossover operator, the inter-individual interactions in the population can be abstracted as complex networks, where the individuals can be represented by the vertices and the interactions among individuals can be denoted by the edges. Now that a crossover operator may be executed between any two individuals in the canonical GA, the interaction structure can be represented by a fully-connected topology.

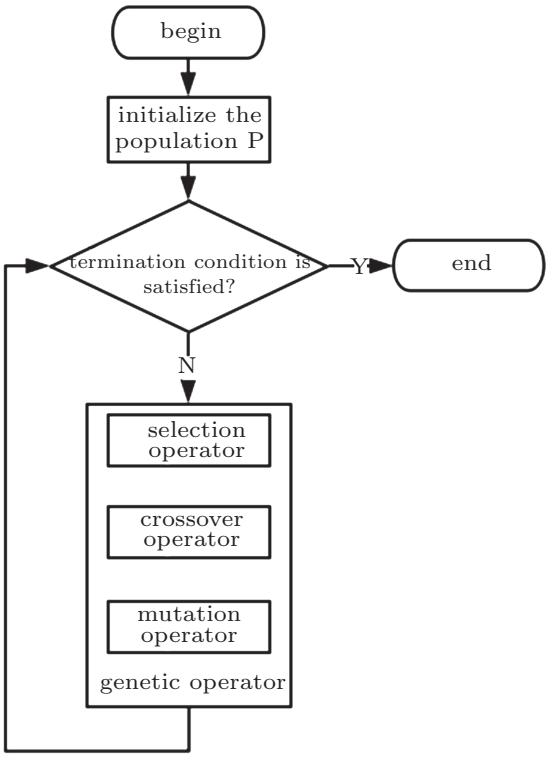


Fig. 1. The flow diagram of genetic algorithm.

In the diagram of Fig. 2, how an offspring generated from their parents in single-point crossover is described.

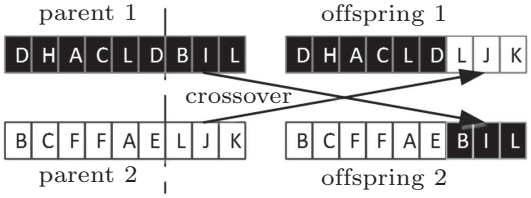


Fig. 2. Crossover operator.

The essence of genetic operators can be explained by the schema theorem proposed by Holland.[24] In GA, a schema is propagated if individuals in the current generation match it and so do those in the next generation. Holland's schema theorem demonstrated that short, low-order schemata with above-average fitness increase exponentially in successive generations. GA is a set of operations for schemata of chromosome in nature. The highly fit schemata are propagated to next generation through the selection operator. Then, the schemata are recombined by crossover operator. In mutation operator, the schemata are mutated to generate new schemata. In conclusion, the schemata evolve to higher fitness value in successive generations, which build the globally optimal solution.

2.2. Benchmark function set

A real-valued function set $F = \{f_1(x), f_2(x), \ldots, f_{15}(x)\}$ consisting of 15 well-known benchmark functions ($f_1 f_{10}$,[25] $f_{11} f_{15}$[26]) is employed to illustrate the performance of t optimization algorithms. These 15 test functions are shown in Table 1.

Table 1. The flow diagram of genetic algorithm.

	Function name	Formula	D	Range$[x_{\min}, x_{\max}]$	Optimum
Unimodal	Sphere	$f_1(x) = \sum_{i=1}^{D} x_i^2$	30	$[-100, 100]^D$	0
	Schwefel 2.22	$f_2(x) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	30	$[-10, 10]^D$	0
	Schwefel 1.2	$f_3(x) = \sum_{i=1}^{D} \left(\sum_{j=1}^{D} x_j \right)^2$	30	$[-100, 100]^D$	0
	Schwefel 2.21	$f_4(x) = \lvert x_i \rvert$	30	$[-100, 100]^D$	0
	Generalized Rosenbrock	$f_5(x) = \sum_{i=1}^{D-1} [100 (x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	$[-30, 30]^D$	0
	Quaric	$f_6(x) = \sum_{i=1}^{D} i x_i^4 + \text{random}[0, 1]$	30	$[-1.28, 1.28]^D$	0
Multimodal	Generalized Rastrigin	$f_7(x) = \sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	$[-5.12, 5.12]^D$	0
	Generalized Griewank	$f_8(x) = \frac{1}{4000} \sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos \frac{x_i}{\sqrt{i}} + 1$	30	$[-600, 600]^D$	0
	Generalized Schwefel 2.26	$f_9(x) = -\sum_{i=1}^{D} x_i \sin \sqrt{\lvert x_i \rvert}$	30	$[-500, 500]^D$	$-418.983D$
	Ackley	$f_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D} \cos(2\pi x_i)\right) + 20 + e$	30	$[-30, 30]^D$	0
	Modified Schwefel	$f_{11}(x) = 418.9829 \times D - \sum_{i=1}^{D} g(z_i)$ $z_i = x_i + 4.209687462275036e+002$ $g(z_i) = \begin{cases} z_i \sin\left(\lvert z_i \rvert^{1/2}\right) & \text{if } \lvert z_i \rvert < 500 \\ (500 - \text{mod}(z_i, 500)) \sin\left(\sqrt{\lvert 500 - \text{mod}(z_i, 500) \rvert}\right) - \frac{(z_i - 500)^2}{10000D} & \text{if } z_i > 500 \\ (\text{mod}(\lvert z_i \rvert, 500) - 500) \sin\left(\sqrt{\lvert \text{mod}(\lvert z_i \rvert, 500) - 500 \rvert}\right) - \frac{(z_i + 500)^2}{10000D} & \text{if } z_i < -500 \end{cases}$	30	$[-100, 100]^D$	0
	HappyCat	$f_{12}(x) = \left\lvert \sum_{i=1}^{D} x_i^2 - D \right\rvert^{1/4} + \frac{0.5\sum_{i=1}^{D} x_i^2 + \sum_{i=1}^{D} x_i}{D} + 5$	30	$[-100, 100]^D$	0
	HGBat	$f_{13}(x) = \left\lvert \left(\sum_{i=1}^{D} x_i^2 \right)^2 - \left(\sum_{i=1}^{D} x_i \right)^2 \right\rvert^{1/2} + \frac{0.5\sum_{i=1}^{D} x_i^2 + \sum_{i=1}^{D} x_i}{D} + 5$	30	$[-100, 100]^D$	0
	Griewank plus Rosenbrock	$f_{14}(x) = f_8(f_5(x_1, x_2)) + f_8(f_5(x_2, x_3)) + \cdots + f_8(f_5(x_{D-1}, x_D)) + f_8(f_5(x_D, x_1))$	30	$[-100, 100]^D$	0
	Scaffer F6	$f_{15}(x) = g(x_1, x_2) + g(x_2, x_3) + \cdots + g(x_{D-1}, x_D) + g(x_D, x_1)$ $g(x, y) = 0.5 + \frac{\sin^2\left(\sqrt{x^2 + y^2}\right) - 0.5}{(1 + 0.001(x^2 + y^2))^2}$	30	$[-100, 100]^D$	0

The first six functions are unimodal functions, and the next nine functions are multimodal functions designed with a considerable amount of local minima. All test functions are tested with the dimension of 30. Stimulations are carried out to find the global minimum of each function.

2.3. The effects of various topologies on the performance of GA

In previously established GAs, the interaction structure is assumed to be a fully-connected topology. It is possible that a crossover operator can be executed between any two individuals in a population. However, the information dissemination velocity depends on the network density, e.g., in the fully-connected topology, each individual's information can be quickly transferred to all other individuals which may cause convergence rapidly with an unsatisfactory solution. Since it was demonstrated that the network structural properties play key roles in PSO algorithm, we are inspired to investigate the effects of various topologies on the performance of GA.

Holland's schema theorem demonstrated that the schemata with high fitness increase exponentially in successive generations, which makes it possible for GA to search the global optimum. Hence, the crossover is the key operator in which short, low-order schemata with above-average fitness are recombined and find the global optimum in the end. We applied the network topology to represent the population interaction structure in GA. A schema matched by an individual represents the information of a vertex in the network. In crossover operator, the recombination of schemata can be abstracted as information dissemination among vertices in the network. The dissemination of schemata-matched global optimum in the network will improve the solution quality. However, the dissemination of schemata-matched local optimum will cause convergence with an unsatisfactory local optimum. Therefore, in a high-density topology, the information dissemination is too fast to escape from the local optimum.

Moreover, the global optimum will be obtained more likely in a low-density topology, though it may impede convergence because of the slow dissemination of information. Hence we use the topologies from the ring to complete connection to test the algorithm performance variation of regular topologies with different network density represented by the average degree (Fig. 3).

In order to investigate the effect of network density, all the topologies above are tested in the framework of RC–GA[11] for the optimization of 15 benchmark functions (Table 1). In RC–GA, the gene consists of real-valued object parameters, and the evolution operates on the natural representation, i.e., each gene is represented by a 64-bit floating point number. As the interaction between two individuals executed mainly on crossover operator, the regular topology is introduced in the step of crossover. The individuals represent the vertices in regular topologies with different network density. Figure 3 displays this feature more intuitively, e.g., in a nearest-neighbor coupled network shown in Fig. 3(b), each individual can crossover with one of 4 neighbors in the population. The modification is also carried out in selection operation, where the offspring's positions in the network do not generate randomly as traditional GA but inherit their parents' positions. Hence, the topologies are maintained in successive generations.

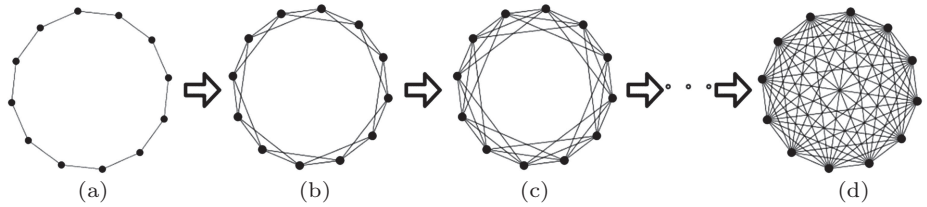


Fig. 3. Networks with different density K (the average degree). The network density grows from left to right. (a) Ring-like network with $K = 2$. (b) Nearest-neighbor coupled network with $K = 4$. (c) nearest-neighbor coupled network with $K = 6$. (d) Fully-connected network.

Table 2 Average and standard deviation of the best fitness found by GA with different network densities.

	Function		Network density									
			$K = 2$	$K = 4$	$K = 6$	$K = 8$	$K = 10$	$K = 20$	$K = 40$	$K = 60$	$K = 80$	$K = 100$
Unimodal	f_1	average	0.307	0.319	***0.294***	0.324	0.333	0.316	0.352	0.299	0.324	0.303
		std. dev	0.728	0.789	0.652	0.688	0.845	0.812	0.730	0.643	0.751	0.749
	f_2	average	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	***0.002***
		std. dev	0.015	0.013	0.013	0.016	0.014	0.015	0.012	0.014	0.018	0.012
	f_3	average	***0.296***	0.332	0.319	0.316	0.330	0.314	0.329	0.339	0.314	0.346
		std. dev	0.591	0.729	0.673	0.734	0.718	0.758	0.906	0.764	0.729	0.795
	f_4	average	1.185	***1.154***	1.193	1.270	1.225	1.234	1.262	1.230	1.209	1.171
		std. dev	1.608	2.085	2.114	2.070	2.012	1.720	2.276	1.946	2.041	1.886
	f_5	average	310.283	314.298	361.973	267.189	552.308	464.098	505.619	682.493	396.968	***261.789***
		std. dev	2256.980	3055.140	3892.500	934.977	9974.810	9413.380	9330.900	11718.100	7084.460	578.992
	f_6	average	0.045	0.047	0.054	0.059	0.034	0.038	0.054	0.039	***0.037***	0.066
		std. dev	0.507	0.457	0.614	0.843	0.333	0.423	1.432	0.377	0.380	1.046
Multimodal	f_7	average	0.117	***0.115***	0.138	0.132	0.130	0.129	0.137	0.128	0.136	0.128
		std. dev	0.261	0.300	0.282	0.278	0.340	0.376	0.316	0.251	0.257	0.353
	f_8	average	***0.852***	0.860	0.874	0.909	0.873	0.860	0.853	0.878	0.861	0.880
		std. dev	1.062	0.815	1.059	0.879	1.088	1.255	1.056	0.880	1.021	0.960
	f_9	average	***−12568.7***	−12568.6	−12568.6	−12568.6	−12568.6	−12568.6	−12568.6	−12568.6	−12568.6	−12568.6
		std. dev	2.128	2.091	2.460	2.380	2.112	2.214	2.029	1.955	1.853	2.127
	f_{10}	average	***0.358***	0.375	0.373	0.392	0.387	0.420	0.414	0.440	0.409	0.416
		std. dev	0.508	0.609	0.514	0.711	0.828	0.780	0.844	0.859	0.827	0.696
	f_{11}	average	***0.0168***	0.0177	0.0185	0.0181	0.0181	0.0171	0.0172	0.0197	0.0174	0.0172
		std. dev	0.033	0.043	0.037	0.038	0.037	0.035	0.040	0.045	0.045	0.037
	f_{12}	average	***2.601***	2.602	2.604	2.604	2.604	2.605	2.604	2.604	2.604	2.603
		std. dev	0.017	0.020	0.025	0.025	0.022	0.022	0.024	0.020	0.027	0.024
	f_{13}	average	0.4925	0.4924	0.4922	0.4919	0.4921	0.4919	***0.4918***	0.4922	0.4920	0.4921
		std. dev	0.008	0.008	0.009	0.010	0.011	0.009	0.010	0.010	0.008	0.010
	f_{14}	average	2.075	***1.934***	2.062	2.020	1.973	2.032	2.056	2.127	2.052	2.017
		std. dev	3.892	3.715	4.604	4.198	4.435	4.316	3.944	4.099	4.429	4.058
	f_{15}	average	1.783	1.779	1.784	1.796	1.717	1.850	1.798	1.949	***1.645***	1.871
		std. dev	4.606	3.915	3.899	4.193	3.671	3.561	3.997	4.142	3.703	4.740

For GA with different topologies, the size of population is set to 100, the probability of crossover and mutation is set to 0.8 and 0.01. Schaffer recommended the range of parameters in GA.[27] Since the algorithms are stochastic, their performance on each test function is evaluated based on statistics obtained from 50 independent runs.

The quality of the averaged optimal fitness is shown in Table 2. The shaded cells in Table 2 indicate that the corresponding algorithm outperforms fully-connected GA ($K = 100$) for a particular function. It is obvious that the shaded cells occupy the most space of the table and concentrate in GAs with low-density topologies, which means that GAs with low-density topologies can obtain better optimal fitness for most of the functions. A value in italic bold indicates that the corresponding algorithm is the best on a particular test function. It can be observed that the fully-connected GA performs the best only for 2 unimodal function (f_2 and f_5), and the GA with ring network ($K = 2$) performs the best for 6 functions. As shown in Table 2, neither the fully-connected topology nor ring topology performs the best for most of the test functions. The information dissemination of fully-connected network is too fast to be able to escape from the local optima, while the ring topology slows the information dissemination and impedes its convergence speed. Therefore, the results confirm the supposition above about the effect of the network density.

2.4. DT-GA model

Theoretically speaking, GA finds the globally optimal solution by obtaining the balance between exploration and exploitation.[28] We find that a low-density topology may benefit exploration while a high-density topology facilitates exploitation. Thus, a dynamic variation from ring network to fully-connected topology may trade off the exploration and exploitation throughout the optimization process, resulting in a good performance. Eiben analyzed adaptive parameter control methods in evolutionary algorithms.[29,30] Srinivas proposed adaptive probabilities of crossover and mutation in genetic algorithms.[12] They insisted that some form of feedback from the search process should be used to determine the direction and/or magnitude of the change to the strategy parameter.

In our algorithm, the network density is modified through the feedback of convergence degree from the search process. To vary network density dynamically, it is essential to identify whether the GA is converging to an optimum. The difference between the average and minimum fitness values, $\Delta = \bar{f} - f_{\min}$, is used as a yardstick for detecting the convergence degree of the GA. In addition, to improve the universality of the algorithm, Δ is normalized according to γ with Eq. (1)

$$\gamma = \frac{\bar{f} - f_{\min}}{\Delta_0}, \tag{1}$$

where Δ_0 represents the difference in the average and minimum fitness values of initial population with Eq. (2)

$$\Delta_0 = \bar{f}_0 - f_{0\min}. \tag{2}$$

Because the convergence degree of initial population is lowest throughout the optimization process, Δ_0 is the maximum of Δs. Then, γ is constrained within the interval [0,1]. γ is likely to be less for a population converged to an optimum solution than that for a population scattered in the solution space. The value of network density K varies along with the value of γ, i.e., K will vary inversely when γ decrease. Moreover, K is an even integer for that the neighbor vertices are symmetric in the nearest-neighbor coupled network. Thus, K can be expressed as shown in Eq. (3)

$$K = \left\lceil \frac{1}{1+\gamma \times c} \times N/2 \right\rceil \times 2, \tag{3}$$

where c is a weight factor applied to γ and N denotes the size of population.

3. Experiments of DT-GA

To evaluate the efficiency of the DT-GA, we compare the optimal fitness obtained from DT-GA with other test GAs whose interaction structures are abstracted as different static network topologies showed in Fig. 3. The results for GAs with different static network topologies are obtained from Section 2. Then, we further explore the dynamical searching process of DT-GA microscopically.

3.1. Experiments setting

For DT-GA, the population sizes are set to 100, the probabilities of crossover and mutation are set to 0.8 and 0.01 respectively. c is set to 50. For each test function, the algorithm executed 50 runs independently. All simulations are performed on a PC with 3.2 GHz CPU and 12 GB of RAM. All GAs are implemented in C++ language.

3.2. Optimization performances

A summary of the results is presented in Table 3, where the shaded cells indicate that the corresponding algorithm outperforms the DT-GA on a particular test function. It can be observed that few test algorithms outperform DT-GA for the functions. In particular, compared to GA with a fully-connected network, DT-GA performs better for 13 test functions. The value in italic bold represents that the corresponding test algorithm performs the best. DT-GA performs the best for 6 test functions. The superiority of DT-GA is apparent. Thus, it is reasonable that DT-GA gives rise to a better balance between the convergence speed and the optimum quality.

Table 3. Average and standard deviation for the best fitness found by test GAs.

	Function		Network density										DT-GA
			$K=2$	$K=4$	$K=6$	$K=8$	$K=10$	$K=20$	$K=40$	$K=60$	$K=80$	$K=100$	
Unimodal	f_1	average	0.307	0.319	***0.294***	0.324	0.333	0.316	0.352	0.299	0.324	0.303	0.302
		std. dev	0.728	0.789	0.652	0.688	0.845	0.812	0.730	0.643	0.751	0.749	0.755
	f_2	average	0.0022	0.0020	0.0019	0.0020	0.0018	0.0020	0.0019	0.0017	0.0023	0.0017	***0.0014***
		std. dev	0.015	0.013	0.013	0.016	0.014	0.015	0.012	0.014	0.018	0.012	0.010
	f_3	average	0.296	0.332	0.319	0.316	0.330	0.314	0.329	0.339	0.314	0.346	***0.296***
		std. dev	0.591	0.729	0.673	0.734	0.718	0.758	0.906	0.764	0.729	0.795	0.625
	f_4	average	1.185	1.154	1.193	1.270	1.225	1.234	1.262	1.230	1.209	1.171	***1.145***
		std. dev	1.608	2.085	2.114	2.070	2.012	1.720	2.276	1.946	2.041	1.886	1.598
	f_5	average	310.283	314.298	361.973	267.189	552.308	464.098	505.619	682.493	396.968	***261.789***	292.312
		std. dev	2256.980	3055.140	3892.500	934.977	9974.810	9413.380	9330.900	11718.100	7084.460	578.992	1007.86
	f_6	average	0.045	0.047	0.054	0.059	0.034	0.038	0.054	0.039	0.037	0.066	***0.014***
		std. dev	0.507	0.457	0.614	0.843	0.333	0.423	1.432	0.377	0.380	1.046	0.113
Multimodal	f_7	average	0.117	***0.115***	0.138	0.132	0.130	0.129	0.137	0.128	0.136	0.128	0.124
		std. dev	0.261	0.300	0.282	0.278	0.340	0.376	0.316	0.251	0.257	0.353	0.291
	f_8	average	***0.852***	0.860	0.874	0.909	0.873	0.860	0.853	0.878	0.861	0.880	0.907
		std. dev	1.062	0.815	1.059	0.879	1.088	1.255	1.056	0.880	1.021	0.960	0.978
	f_9	average	***−12568.7***	−12568.6	−12568.6	−12568.6	−12568.6	−12568.6	−12568.6	−12568.6	−12568.6	−12568.6	***12568.7***
		std. dev	2.128	2.091	2.460	2.380	2.112	2.214	2.029	1.955	1.853	2.127	1.941
	f_{10}	average	0.358	0.375	0.373	0.392	0.387	0.420	0.414	0.440	0.409	0.416	***0.342***
		std. dev	0.508	0.609	0.514	0.711	0.828	0.780	0.844	0.859	0.827	0.696	0.586
	f_{11}	average	***0.0168***	0.0177	0.0185	0.0181	0.0181	0.0171	0.0172	0.0197	0.0174	0.0172	0.0169
		std. dev	0.033	0.043	0.037	0.038	0.037	0.035	0.040	0.045	0.045	0.037	0.041
	f_{12}	average	***2.601***	2.602	2.604	2.604	2.604	2.605	2.604	2.604	2.604	2.603	2.602
		std. dev	0.017	0.020	0.025	0.025	0.022	0.022	0.024	0.020	0.027	0.024	0.022
	f_{13}	average	0.4925	0.4924	0.4922	0.4919	0.4921	0.4919	***0.4918***	0.4922	0.4920	0.4921	0.492
		std. dev	0.008	0.008	0.009	0.010	0.011	0.009	0.010	0.010	0.008	0.010	0.009
	f_{14}	average	2.075	***1.934***	2.062	2.020	1.973	2.032	2.056	2.127	2.052	2.017	1.940
		std. dev	3.892	3.715	4.604	4.198	4.435	4.316	3.944	4.099	4.429	4.058	3.649
	f_{15}	average	1.783	1.779	1.784	1.796	1.717	1.850	1.798	1.949	***1.645***	1.871	1.735
		std. dev	4.606	3.915	3.899	4.193	3.671	3.561	3.997	4.142	3.703	4.740	4.229

3.3. Search process of DT-GA

The essence of the algorithm can be illustrated by Fig. 4 which shows the variation of minimum fitness value ($f_{\min}$), convergence degree (γ) and network density (K) during search process.

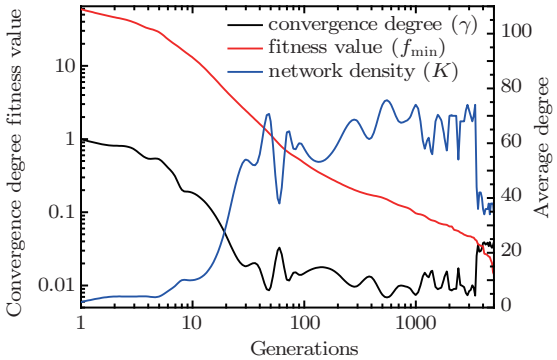


Fig. 4. (color online) Variation of $f_{\min}$, γ, and K.

At the beginning of evolution, the network density is set to 2 considering the information dissemination of the ring network ($K=2$) is so slow that GA has enough time to find a better optimum. When the convergence speed slows down, the convergence degree decreases and K increases inversely since a high-density network can speed up convergence. Therefore, the negative correlation between convergence degree and network density makes DT-GA trade off the exploration and exploitation throughout the optimization process

4. Conclusion and perspectives

In this paper, we have investigated the effects of various topologies on the performance of GA and proposed an improved GA with dynamic topology. First, to evaluate the effects of topologies with different network density on the performance of GA, the topologies from ring to fully-connected have been used to represent the interaction structure in GA. It is convinced that GAs high-density topology are more likely to give out an imperfect solution, while a low-density topology may prevent convergence. Then, we have proposed a GA with dynamic topology, named DT-GA. To demonstrate the efficiency of the method, the algorithm has been evaluated using 15 famous benchmark functions. The results have illus-

trated that DT-GA gives rise to a better balance between the convergence speed and the optimum quality. Nonetheless, our algorithm can be further improved in the convergence speed. In future study, the evolutionary algorithm will be introduced in other complex network with shorter average path length and larger clustering coefficient, e.g., the small-world network and the scale-free network.

Acknowledgment

We thank Prof. Wen-Bo Du for his helpful discussions.

References

[1] Zwickl D J 2006 "Genetic Algorithm Approaches for the Phylogenetic Analysis of Large Biological Sequence Datasets Under the Maximum Likelihood Criterion," Ph. D. Dissertation, Austin, University of Texas at Austin
[2] Poli R, Kennedy J and Blackwell T 2007 *Swarm Intell.* **1** 33
[3] Dorigo M and Blum C 2005 *Theor. Comput. Sci.* **344** 243
[4] Holland J H 1975 *Adaptation in Natural and Artificial Systems* (Ann Arbor: University of Michigan Press) pp. 171–198
[5] Zhang S L, Chen H S, Song Y and Yin Y H 2007 *Acta Phys. Sin.* **56** 2553 (in Chinese)
[6] Li T J Sun Y, Zheng J W, Shao G F and Liu T D 2010 *Acta Phys. Sin.* **64** 153601 (in Chinese)
[7] Zu Y X, Zhou J and Zeng C C 2010 *Chin. Phys. B* **19** 119501
[8] Lu J, Wang J B and Sun G C 2009 *Chin. Phys. B* **18** 1598
[9] Hollstein R B 1971 "Artifical Genetic Adaptation in Computer Control Systems", Ph. D. Dissertation, Ann Arbor, University of Michigan
[10] Goldberg D E 1983 "Computer-aided Gas Pipeline Operation Using Genetic Algorithms and Rule Learning", Ph. D. Dissertation, Ann Arbor, University of Michigan
[11] Goldberg D E 1991 *Complex Syst.* **5** 139
[12] Srinivas M and Patnaik L M 1994 *IEEE Trans. Syst. Man Cybern.* **24** 656
[13] Deb K, Pratap A, Agarwal S and Meyarivan T 2002 *IEEE Trans. Evol. Comput.* **6** 18
[14] Zhang Q and Li H 2007 *IEEE Trans. Evol. Comput.* **11** 712
[15] Szabó G and Fáth G 2006 *Phys. Rep.* **446** 97
[16] Yang H X, Gao K, Han X P and Wang B H 2008 *Chin. Phys. B* **17** 2759
[17] Strogatz S H 2001 *Nature* **410** 268
[18] Xiong F, Liu Y, Si X M and Ding F 2010 *Acta Phys. Sin.* **59** 6889 (in Chinese)
[19] Xing C M and Liu F A 2010 *Acta Phys. Sin.* **59** 1608 (in Chinese)
[20] Liu C, Du W B and Wang W X 2014 *PLoS One* **9** e97822
[21] Du W B, Gao Y, Liu C, Zheng Z and Wang Z 2015 *Appl. Math. Comput.* **268** 832
[22] Gao Y, Du W B and Yan G 2015 *Sci. Rep.* **5** 9295
[23] Du W B, Ying W, Yan G, Zhu Y B and Cao X B 2016 *IEEE Trans. Circuits Syst. II: Exp. Briefs*
[24] Negvitsky M 2005 *Artificial Intelligence*, 2nd edn. (Essex: Pearson Ed. Ltd) pp. 1–415
[25] Yao X, Liu Y and Lin G 1999 *IEEE Trans. Evol. Comput.* **3** 82
[26] Liang J J, Qu B Y and Suganthan P N 2013 Technical Report, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore
[27] Schaffer J D, Caruana R A, Eshelman L J and Das R 1989 *Proceedings of the 2nd International Conference on Genetic Algorithms*, June, Fairfax, USA, p. 51
[28] Tan K C, Chiam S C, Mamun A A and Goh C K 2009 *European Journal of Operational Research* **197** 701
[29] Eiben A E, Hinterding R and Michalewicz Z 1999 *IEEE Trans. Evol. Comput.* **3** 124
[30] Karafotias G, Hoogendoorn M and Eiben A E 2015 *IEEE Trans. Evol. Comput.* **19** 167