Applied Mathematics and Computation 338 (2018) 33–43

Contents lists available at ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

The networked evolutionary algorithm: A network science perspective

Wenbo Du[a], Mingyuan Zhang[a], Wen Ying[a], Matjaž Perc[a,b,*], Ke Tang[c], Xianbin Cao[a,**], Dapeng Wu[d,**]

[a] School of Electronic and Information Engineering, National Engineering Laboratory for Big Data Application Technologies for Comprehensive Traffic, Beihang University, Beijing 100191, China
[b] Faculty of Natural Sciences and Mathematics, University of Maribor, Koroška cesta 160, Maribor SI-2000, Slovenia
[c] Shenzhen Key Laboratory of Computational Intelligence, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, Guangdong 518055, China
[d] Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611, USA

ARTICLE INFO

Keywords:
Evolutionary algorithm
Network system
Structure
Behavior

ABSTRACT

The evolutionary algorithm is one of the most popular and effective methods to solve complex non-convex optimization problems in different areas of research. In this paper, we systematically explore the evolutionary algorithm as a networked interaction system, where nodes represent information process units and connections denote information transmission links. Within this networked evolutionary algorithm framework, we analyze the effects of structure and information fusion strategies, and further implement it in three typical evolutionary algorithms, namely in the genetic algorithm, the particle swarm optimization algorithm, and in the differential evolution algorithm. Our results demonstrate that the networked evolutionary algorithm framework can significantly improve the performance of these evolutionary algorithms. Our work bridges two traditionally separate areas, evolutionary algorithms and network science, in the hope that it promotes the development of both.

1. Introduction

Optimization is crucial in scientific research and engineering areas [1–3]. Many real-world problems can be formalized as minimization optimization as follow:

$$\begin{aligned} \min \quad & f(\mathbf{x}), && \mathbf{x} = [x_1, x_2, \ldots, x_D] \\ \text{s.t.} \quad & g_i(\mathbf{x}) \le 0, && i = 1, 2, \ldots, n, \end{aligned} \tag{1}$$

where D is the dimension of the problem, n is the number of constraints, $f(\mathbf{x})$ is an objective function and $g_i(\mathbf{x})$ are constraint conditions. As many real-world problems are NP hard and grow increasingly complex, the corresponding objective functions are generally non-convex, with non-smooth surfaces and numerous local optima, where most of the gradient-

* Corresponding author at: School of Electronic and Information Engineering, National Engineering Laboratory for Big Data Application Technologies for Comprehensive Traffic, Beihang University, Beijing 100191, China.
** Corresponding authors.
E-mail addresses: matjaz.perc@uni-mb.si (M. Perc), xbcao@buaa.edu.cn (X. Cao), dpwu@ufl.edu (D. Wu).

https://doi.org/10.1016/j.amc.2018.06.002
0096-3003/© 2018 Elsevier Inc. All rights reserved.

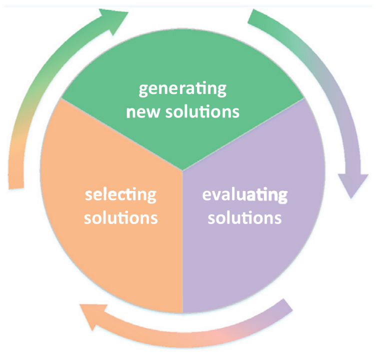

Fig. 1. Framework of evolutionary algorithms.

based convex optimization methods, such as gradient descent Newton–Raphson method, are incompetent. Hence tremendous efforts have been dedicated to developing more intelligent algorithms to handle multimodal complex optimization problems.

An evolutionary algorithm (EA) is inspired by natural or human intelligence [4]. In an evolutionary algorithm, a solution of the objective function is mapped as an individual, and the value of the objective function corresponds to the evaluation of the individual. Furthermore, a group of solutions are reserved simultaneously, known as individuals in a population. A batch of new solutions is generated in each iteration via crossover and mutation operators, and then some of the existing solutions and new solutions will be reserved by selection mechanisms and get in the next iteration. As the algorithm runs iteratively, the objective function is searched randomly but with a certain trend determined by the algorithm, and the quality of solutions will gradually increase. In the past decades, several EAs have been proposed to improve the performance on optimization problems, such as genetic algorithm [5], particle swarm optimization [6], differential evolution [7], and the artificial bee colony algorithm [8], to name just some (we refer to [9] for a Python-based microframework for building nature-inspired algorithms). These algorithms look very different with various descriptions and mechanisms, yet they share the same framework in form: generating new solutions, evaluating solutions, selecting solutions, as shown in Fig. 1. Essentially, all the EAs devote to make a tradeoff between exploration and exploitation, i.e. how widely to search in the global space and how precisely to dig around the local space.

Information interaction, i.e. communication between individuals, is crucial for EAs. As most of the real-world optimization problem is continuous and slow-varying, a high-quality solution (a position in the search space) that has been found usually indicates a promising region, i.e. it is more probable to find high-quality solutions around this region. From that point of view, the optimization process is similar to the process of gold mining. Suppose that two teams with a same number of miners compete for gold mining. Team *A* is well organized with a good communication mechanism while team *B* is a group of disorganized miners. Obviously, team *A* is more efficient, because team *A* is able to spread information about where gold has been found, yet in team *B*, each person could only utilize self-experience.

Furthermore, two aspects are significant to team *A*'s effectiveness of the mining process. The first is structure. The team with fully-connected structure is usually not a good choice, as over-redundant information may interfere one's judgement. Yet an over-sparse structure is inefficient for spreading information. A preferable scheme should be with asymmetric structure where numerous miners could only communicate with several hub miners yet not all miners. The second is information processing strategy. Based on the asymmetric structure above, the structural importance of miners are quite different. Though all miners could follow the same behavior pattern, it would be more efficient if the hub miners become leaders, because hub miners could collect more information thus have better understandings of the mineral distribution.

Similarly, a well-designed EA should also take into account the importance of structural and behavioral heterogeneity. However, these are in fact not well-implemented in most current EAs. Numerous information processing strategies have been proposed to improve the performance, some of which could achieve remarkable results, yet individuals are designed with the same pattern due to the intrinsic homogeneity of these systems [10]. A few works notice the importance of structure, however, most of them still focus on distance-based cluster structure [11,12], multiswarm technique [13,14] or regular structures [15,16]. Hence most EA variants are essentially limited in balancing exploration and exploitation.

In this paper, we consider the EA as a networked information transmission system (NITS). To improve the performance, we employ heterogeneous network structure and design appropriate information fusion strategies of nodes (individuals). Structurally important nodes, i.e. hub nodes, mainly play the role of processing and distributing information. Non-hub nodes provide their information to the hubs while not always follow the hubs' guide, but instead do explorations themselves to find other promising regions sometimes. Based on this idea, we propose the networked evolutionary algorithm (NEA) framework, exploring the effects of population structure on the performance. In the past decades, the advances in network science have shown that structure plays an important role in the functionalities of a system, such as robustness, spreading, cooperation, synchronization, controllability, and so on. It is fairly expected that the population structure will also impact the performance of evolutionary algorithm. Indeed, we implement NEA on three representative and most popular EAs: genetic algorithm (GA) [5], particle swarm optimization (PSO) [6] and differential evolution (DE) [7], finding that the function of EA system can be remarkably enhanced via modifying structure and designing behavior.

The rest of this paper is organized as follows. Section 2 describes the NEA framework in detail, including the discussion about the relationship between EA and NITS, and the design principles of NEA. Sections 3 to 5, respectively present the implements of the NEA framework in three popular EAs: genetic algorithm, particle swarm optimization and differential evolution. A conclusion is made in Section 6.

2. Networked evolutionary algorithm framework

2.1. From EA to NITS

In NITS, the following aspects have drawn most attentions:

- network structure, where nodes represent information process units and connections denote information transmission links;
- network information transmission, which determines how a node communicates with its neighbors;
- network dynamics, which concerns about how a node utilizes its obtained information.

For EA, network structure is specifically the population structure, describing the connection relationship between individuals. Information transmission refers in particular how an individual obtains information from its neighbors. For instance, each individual randomly selects one neighbor as information source in the original genetic algorithm [5], while in fully-informed particle swarm optimization [17], each particle receives information from all its neighbors. Network dynamics is about how an individual fuses information and generates new solutions, which is presented differently in different algorithms, such as crossover in genetic algorithm [5], velocity update formula in particle swarm optimization [6]. As both information transmission and network dynamics are the actions of individuals to handle information, we unify them as the individual behaviors (information fusion strategies) in EA.

In most systems, structure and behavior are interrelated and interacted [18–20]. The structure usually constrains behavior, as only the node pairs with connections could exchange information. Meanwhile, behavior provides feedback to structure. Since only information from partial neighbors are utilized for a certain node, not all existing connections are effective or even necessary to provide information to the node. Over-redundant connections may even reduce the functionality. In addition, a system should better employ nodes with diverse behaviors to face various situations for complex tasks. Thus, the feedback of behavior to structure might be various and heterogeneous. Therefore, we propose the networked evolutionary algorithm (NEA), which should be designed by the two following rules:

- choosing a relatively sparse yet efficient network structure for spreading information;
- designing appropriate information fusion strategies for different roles of individuals based on structure.

2.2. The design principle of NEA

The function of a NEA system is to achieve better performance on optimization problems. The performance is usually determined by how well this system balances exploration and exploitation. This balance is implemented by coordinatingly designing population structure and individual behaviors. In most previous studies of EAs, fully-connected or ring structure are widely employed, thus individuals are homogeneous from the point of view of topology, hardly avoiding sharing the same pattern. In NEA, we employ heterogeneous structure, which provides differentiated topological information. Hence, the diversity of individual behaviors could naturally be designed based on the heterogeneous topology.

Specifically, communication network structure is the concept of population structure in NEA. In NEA, each individual is treated as a node in the communication network system, while the connection between two nodes refers to that both nodes could obtain information from each other via a certain way. Population structure has been studied in some of EAs, such as PSO. Yet, in other algorithms, like GA, the impact of population structure has not been systematically explored. In the NEA framework, we employ the BA network as the structure, which is proposed by Barabási and Albert in 1999 and becomes one of the most representative heterogeneous structure [21]. There are three advantages of employing BA model:

- Most of nodes in a BA network are relatively low-degree, thus avoiding processing massive yet valueless information;
- The average distance of BA network is small, implying that the high-quality information could be effectively and rapidly spread in the network;
- The heterogeneous topology, as mentioned above, helps in establishing heterogeneous information fusion strategies.

The generation process of BA network is as follows. The network starts from a fully-connected core with a small number m_0 of nodes, and at each step a new node is added and connected with m existing nodes. The probability that the new node connects to an existing node i is proportional to i's degree.

The information fusion strategy (behavior) is very different between specific algorithms. In the canonical PSO [6], for instance, each individual (particle) extracts the information of the neighbor with best fitness, and combines with the information of itself to create a new position; while in GA [5], the position information is encoded as genotypes, and each individual creates new information via non-linearly recombining its genotype with a random neighbor. Moreover, numerous variants of EAs use information fusion strategies that are more complicated [10,17,22,23]. Hence, in this section we mainly introduce the design principles. In NEA framework, we design heterogeneous information fusion strategies for individuals mostly based on the heterogeneous topology. The structurally-important hub nodes are encouraged to obtain information as much as possible. To some extent, these hubs work as leaders. Non-hub nodes are designed to obtain information from their neighbors in probability. As most non-hub nodes directly connect with hubs in a BA network, they could obtain high-quality information from these hubs. Furthermore, since non-hub nodes have fewer neighbors, they communicate with the hub neighbor more frequently. Meanwhile, the non-hub nodes have a probability to explore new regions driven by itself or other non-hub neighbors, hence the system could avoid being completely controlled by several hub nodes. The next three sections show in detail how we redesign the information fusion strategies in three representative and typical EAs: GA, PSO and DE.

3. NEA in genetic algorithm

3.1. Analysis of genetic algorithm

Genetic Algorithm (GA) [5], proposed by Holland and his colleagues in 1970s, is one of the most famous and widely-used EA. Inspired by Darwin's theory of evolution, GA represents the solution information of optimization problem as genes of an individual. In each generation, the population generate offsprings by crossover operators between individual pairs and mutation operators on each single individual. Then a selection operator is executed on the whole population with both parents and offsprings to maintain the size of population, where individuals with better fitness are more likely to survive into the next generation. By repeating these operators, the fitness of the whole population will be improved gradually.

Among the evolution process above, we are most interested in the crossover operator, because it is the main part of information fusion occurring in GA. In classical GA, the crossover partner of each individual is randomly selected in the whole population. From the NITS point of view, it means that each individual may get information from all the other individuals, hence the structure of classical GA can be considered as a fully connected network. In each generation, each individual achieves information fusion by receiving information from all its neighbors then randomly picking one to cross with. Note that the individuals are changing along with evolution process, hence the fully connected structure of the population is actually varying in a dynamic way, i.e. dead individuals and their links are removed while the survived newborns are connected with other survived individuals.

The selection operator determines that the high-quality individuals (i.e. the individual with good fitness) are more probable to survive. Therefore, high-quality information in the solution space will be preserved in the population. Meanwhile, the survived high-quality individuals could constantly spread their information in population to further help improve other individuals.

3.2. Networked genetic algorithm

Based on NEA framework and the analysis of classical GA, we propose the networked genetic algorithm (N-GA).

Firstly, we employed BA network, instead of fully connected network, as the population structure. Besides the two advantages for all NEAs, the growth property of BA network particularly accord with the dynamic structure of GA.

In N-GA, each individual randomly selects a crossover partner from its neighbors. Yet due to the structural heterogeneity, the high-degree hub individuals will be chosen more frequently than the non-hub. Note that, in N-GA, since each individual preforms crossover with probability Pc, the total number of offsprings generated by the crossover operator will be about $2^{*}N^{*}Pc$, where N is the population size.

Survived offsprings join the population by following the generation process of BA model. Specifically, the offspring will connect to m random existing individuals in the network, an the probability of connecting to individual i is proportional to i's degree.

As high-quality individuals are more likely to survive, they are expected to have more neighbors. Consequently, they have more chances to spread their high-quality information to the whole population.

Fig. 2 shows an example with $N = 6$. Fig. 2 (a) and (b) illustrate the process of generating offsprings and evaluating individuals. Different colors represent different genes. The stars on each individual represent the solution quality. Specifically, the individuals with more black stars own high-quality solutions. And the arrows from Fig. 2 (a) to 2 (b) indicate the copy of genes from parents to offsprings, i.e. the information flow in GA. The links in Fig. 2 (a) imply the neighborhood relationship of two individuals. Note that each individual would choose one of its neighbors to generate a child. Fig. 2 (c) shows the process of selection, where the survived individuals are denoted by dashed boxes and the dead individuals are marked by a sad face. When a parent individual is going to die, its links will also be removed, denoted as transparent links in Fig. 2 (c). Fig. 2 (d) displays the process of connecting the offsprings into the population following the preferential attachment mechanism in BA model, denoted as dash lines. Note that the links will be inherited if both ends of the parent individuals are survived.

The other operators (mechanisms) and parameters used in N-GA and GA are listed in Table 1 (following [24]).

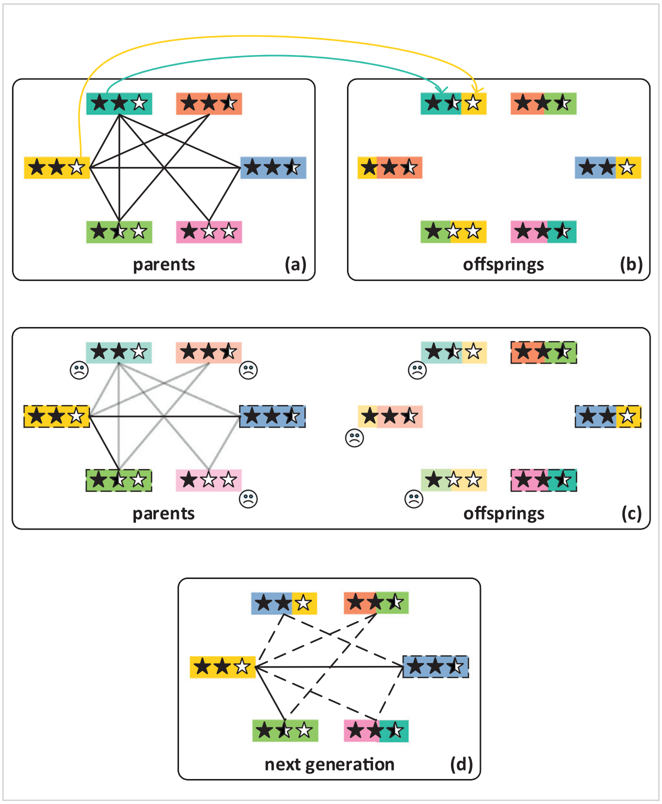

Fig. 2. Schematic diagram of the networked genetic algorithm.

Table 1
The operators and parameters setting in GA and N-GA.

Operators and parameters	Description (Value)
Crossover mechanism	Single point crossover
Crossover probability	0.95
Mutation mechanism	Single point mutation
Mutation probability	0.1
Selection mechanism	Roulette wheel selection
Population size	125
m_0 of BA network	4
m of BA network	4

3.3. *Experiments*

We evaluate the performance of N-GA with 7 widely-used benchmark functions [24,25]. The formula and other details of these problems are listed in Table 2, where f_1 and f_2 are unimodal while $f_3 - f_7$ are multimodal. The dimension of all problems is set as 30.

To compare with N-GA, we employ classical GA with the same parameters yet fully connected structure following Table 1. Table 3 shows the average fitness values of 50 independent runs found by N-GA and GA. The max iteration of each run is set to 2000.

As shown in Table 3, N-GA outperforms GA on all of 7 problems. The results demonstrate the power of NEA framework.

4. NEA in particle swarm optimization

4.1. *Analysis of particle swarm optimization*

Particle swarm optimization (PSO) is proposed by Kennedy and Eberhart in 1995 [6], imitating the behavior of bird flocking and fish schooling. In PSO, a population of N particles fly in the D dimensional solution space, where each position, denoted as a vector $\mathbf{x} = [x_1, x_2, \dots, x_d, \dots, x_D]$, in the space represents a solution of the objective function. Each particle i

Table 2
Benchmark functions for genetic algorithm.

Formula	Range
$f_1(x) = -20exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2})$	
$-exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} cos2\pi x_i}) + 20 + e$	$[-32, 32]^D$
$f_2(x) = \sum_{i=1}^{D} x_i{}^2$	$[-100, 100]^D$
$f_3(x) = \sum_{i=1}^{D} \lvert x_i\rvert + \prod_{i=1}^{D} \lvert x_i\rvert$	$[-10, 10]^D$
$f_4(x) = \sum_{i=1}^{D} ix_i^4 + random[0, 1)$	$[-1.28, 1.28]^D$
$f_5(x) = \sum_{i=1}^{D} x_i{}^2 - 10\cos 2\pi x_i + 10$	$[-5.12, 5.12]^D$
$f_6(x) = \frac{1}{4000}\sum_{i=1}^{D} x_i{}^2 - \prod_{i=1}^{D}\cos\frac{x_i}{\sqrt{i}} + 1$	$[-600, 600]^D$
$f_7(x) = \sum_{i=1}^{D-1} 100(x_{i+1} - x_i{}^2)^2 + (x_i - 1)^2$	$[-32, 32]^D$

Table 3
Results of average fitness value.

Functions	GA	N-GA
f_1	$3.18E02$	**2.35E02**
f_2	$1.77E01$	**1.68E00**
f_3	$4.53E-01$	**2.52E-01**
f_4	$1.01E-01$	**6.47E-02**
f_5	$2.85E01$	**2.16E01**
f_6	$1.02E00$	**6.30E-01**
f_7	$2.45E00$	**1.94E00**

learns from good experiences of itself or other particles, and updates its velocity $\mathbf{v}_i = [v_{i,1}, v_{i,2}, \ldots, v_{i,d}, \ldots, v_{i,D}]$ and position $\mathbf{x}_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,d}, \ldots, x_{i,D}]$ iteratively with the following equations:

$$x_{i,d} := x_{i,d} + v_{i,d} \tag{2}$$

$$v_{i,d} := v_{i_d} + c_1 * r_{1,i,d} * (pbest_{i,d} - x_{i,d}) \\ + c_2 * r_{2,i,d} * (gbest_d - x_{i,d}), \tag{3}$$

where $\mathbf{pbest}_i = [pbest_{i,1}, pbest_{i,2}, \ldots, pbest_{i,d}, \ldots, pbest_{i,D}]$ is the best position (position with best fitness) that i has experienced from the beginning till now; $\mathbf{gbest} = [gbest_1, gbest_2, \ldots, gbest_d, \ldots, gbest_D]$ is the best position among all **pbest**s; $c1$ and $c2$ are constant coefficients; $\mathbf{r}_{1,i} = [r_{1,i,1}, r_{1,i,2}, \ldots, r_{1,i,d}, \ldots, r_{1,i,D}]$ and $\mathbf{r}_{2,i} = [r_{2,i,1}, r_{2,i,2}, \ldots, r_{2,i,d}, \ldots, r_{2,i,D}]$ are D dimensional vectors and each dimension is a random number in the uniform distribution $U[0, 1]$. In each iteration, each particle i examines whether the fitness of a new position $\mathbf{x}_i$ is better than the fitness of $\mathbf{pbest}_i$, if so, $\mathbf{pbest}_i$ will be replaced by the new $\mathbf{x}_i$.

The mechanism of PSO is simple yet effective, where the velocity equation plays an important role, not only because the velocity equation determines the movement of each particle then further determines the performance of PSO algorithms, but it also essentially determines the information fusion strategy of PSO. For instance, in the classical PSO introduced above, particles utilize the information of **pbest**s and **gbest** when updating velocity. In other words, **pbest**s and **gbest** act as information sources that provide the information of the solution space to the whole population. Note that all particles learn from the best position of all **pbest**s, i.e., the population is fully connected, where each particle could directly use the information of any other one. While in some variants of PSO, several other structures, such as a ring [15] or hierarchies [16], are employed. In these variants, **gbest** is replaced by $\mathbf{nbest}_i$, which is the best position of i's neighbors, and particles may utilize different information from the population rather than only the global best one. Furthermore, besides learning from itself, each particle could learn from more than one information source simultaneously. In FIPSO, all neighbors can be information sources [17]. In CLPSO, each particle may utilize different information sources in different dimensions [26]. Studies of all these variants enrich the design of information fusion strategies in PSO, yet most of them still homogenize particles. We also refer to [27] for related PSO algorithms.

A significant feature that distincts PSO from other popular EAs is that particles (individuals) have memories about the solution space. Specifically, each particle remembers the best position it has ever reached before. When a particle updates its velocity, the information actually flows from these memory positions to the particle [13]. As low-quality $\mathbf{pbest}_i$ will easily be replaced by a new high-quality position, the quality of memory positions will be improved, providing high-quality information to particles.

4.2. Networked particle swarm optimization

In this section, we generalize the comprehensive learning particle swarm optimization and propose networked PSO (N-PSO) as an instance to show how NEA framework works for PSO.

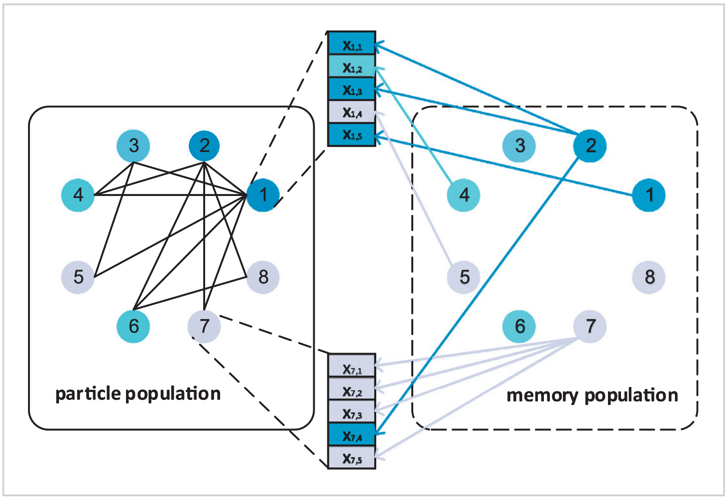

Fig. 3. Schematic diagram of the networked particle swarm optimization.

In N-PSO, a BA network is used to represent the population structure. As most particle swarm optimizations suffer from premature convergence due to the overabundant connections, a sparse BA network could limit the spreading speed of information thus overcome premature convergence to a certain extent [28].

Each particle in N-PSO updates its velocity [26] with the following equation:

$$v_{i,d} := w_g * v_{i,d} + c * r_{i,d} * (pbest_{f_{i,d},d} - x_{i,d}) \tag{4}$$

$$w_g = w_0 - \frac{(w_0 - w_1) * g}{g_{\max}}, \tag{5}$$

where w_g is the inertia weight varing from w_0 to w_1; g is the current iteration and $g_{\max}$ is the pre-set number of total iterations. $\mathbf{pbest}_{f_{i,d}}$=[$pbest_{f_{i,d},1}$, $pbest_{f_{i,d},2}$, ..., $pbest_{f_{i,d},d}$, ..., $pbest_{f_{i,d},D}$] is a constructed **pbest** vector, where $pbest_{f_{i,d}}$ represents the dth dimension of particle $f_{i,d}$'s **pbest**. Note that for each dimension, $f_{i,d}$ could be different. For each dimension of each particle, $f_{i,d}$ is determined as follow: Particle i will randomly decide to learn from its neighbor with a learning probability pl_i, otherwise $f_{i,d}$ is i; If it decides to learn from its neighbors, two neighbors will be randomly selected and $f_{i,d}$ is the one with a better fitness.

Another important factor in PSO is pl. In N-PSO, pl is associated with particle's degree as follow:

$$pl_i = \frac{1}{1 + e^{2*\bar{k} - k_i}}, \tag{6}$$

where k_i is i's degree and $\bar{k}$ is the average degree of the population. Based on Eq. (6), hubs will be more likely to learn from neighbors and utilize more information for self-improving, thus hubs could provide high-quality information. Meanwhile, non-hubs tend to learn themselves in most times, but they could obtain high-quality information more easily when deciding to learn from neighbors.

For instance, in Fig. 3, the population structure is a BA network with $m_0 = m = 2$ and black lines represent the neighborhood relationship of two particles. Each particle has 5 dimensions where for particle 1 and 7, the vector of 5 dimensions are expanded for a better explanation. The colored arrows implies the information flow (which the particle learns from in the specific dimension). Note that another part of information flow that from the particle population to the memory population when updating **pbest** is not displayed in Fig. 3. Hub particle 1 learns from particle 2 for dimension 1, i.e. $f_{i,d} = 2$ in this case, and for other dimensions, it is similar. Hence for particle 1, $\mathbf{pbest}_{f_{i,d}}$=[2, 4, 2, 5, 1]. While for non-hub particle 7, $\mathbf{pbest}_{f_{i,d}}$ = [7, 7, 7, 2, 7].

4.3. Experiments

We adopt benchmark problems in Table 4 to evaluate the performance of N-PSO, in comparisons with PSO [29] and CLPSO [26], where the first three problems are unimodal and the rest are multimodal [25,26]. The number of dimensions of all problems are set as 30. Based on previous experiences [26,28–30], we set $w = 0.729$ and $c1 = c2 = 1.494$ in PSO; in CLPSO and N-PSO, $w_0 = 0.9$, $w_1 = 0.4$, $c = 1.494$ and the refresh gap is set as 7; the parameters of the BA network in N-PSO are set as $m_0 = m = 4$. The comparison results are shown in Table 5, where each result is the average of 30 independent runs. The max iteration of each run is set as 5000.

Table 4
Benchmark functions for particle swarm optimization.

Formula	Range
$f_1(x)=\sum_{i=1}^{D} x_i{}^2$	$[-100,100]^D$
$f_2(x)=\sum_{i=1}^{D-1} 100(x_{i+1}-x_i{}^2)^2+(x_i-1)^2$	$[-32,32]^D$
$f_3(x)=\sum_{i=1}^{D} ix_i^4+random[0,1)$	$[-1.28,1.28]^D$
$f_4(x)=-20exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2})+20$	
$-exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}cos2\pi x_i})+e$	$[-32,32]^D$
$f_5(x)=418.9829\times D+\sum_{i=1}^{D} x_i\sin(\lvert x_i\rvert^{\frac{1}{2}})$	$[-500,500]^D$
$f_6(x)=\sum_{i=1}^{D} x_i{}^2-10\cos 2\pi x_i+10$	$[-5.12,5.12]^D$
$f_7(x)=\sum_{i=1}^{D} y_i{}^2-10\cos(2\pi y_i)+10$	
$y_i=\begin{cases} x_i & \lvert x_i\rvert<\frac{1}{2} \\ \frac{round(2x_i)}{2} & \lvert x_i\rvert\ge\frac{1}{2}\end{cases}$	$[-0.5,0.5]^D$
$f_8(x)=\sum_{i=1}^{D}(\sum_{k=0}^{kmax}[a^k\cos(2\pi b^k(x_i+0.5))])$ $-D\times\sum_{k=0}^{kmax}[a^k\cos(2\pi b^k)]$	
$a=0.5, b=3, kmax=20$	$[-0.5,0.5]^D$
$f_9(x)=\frac{1}{4000}\sum_{i=1}^{D} x_i{}^2-\prod_{i=1}^{D}\cos\frac{x_i}{\sqrt{i}}+1$	$[-600,600]^D$

Table 5
Results of average fitness value.

Functions	PSO	CLPSO	N-PSO
f_1	**2.60E-92**	$3.57E-15$	$8.70E-20$
f_2	**1.02E01**	$1.69E01$	$2.12E01$
f_3	$3.24E-03$	$5.63E-03$	**2.47E-03**
f_4	$1.34E00$	$1.49E-08$	**1.66E-10**
f_5	$5.15E03$	$3.82E-04$	**3.82E-04**
f_6	$5.47E01$	$4.71E-07$	**2.39E-08**
f_7	$3.60E01$	$6.28E-05$	**3.03E-06**
f_8	$5.37E00$	$2.39E-08$	**3.30E-11**
f_9	$1.11E-02$	$1.44E-12$	**3.22E-15**

As shown in Table 5, N-PSO wins on f_3 to f_9. While CLPSO is known to be good at multimodal problems, N-PSO further outperforms on all of multimodal problems. N-PSO also performs better than CLPSO on unimodal problems. Only on f_2 N-PSO is only beaten by CLPSO on f_2, demonstrating the effectiveness of NEA framework.

5. NEA in differential evolution

5.1. Analysis of differential evolution algorithm

Differential evolution (DE) algorithm, first proposed by R. Stone and K. V. Price in 1995, is another typical and competitive evolutionary algorithm [7]. In DE, each individual i, denoted as $\mathbf{x}_i=[x_{i,1},x_{i,2},\dots,x_{i,d},\dots,x_{i,D}]$, presents a potential solution with D variables. As an evolution algorithm, a population of N individuals will be initialized in the solution space, then in each iteration, the three following operators will be executed on each individual i:

- Mutation: Mutation in DE creates a temp solution $\mathbf{v}_i=[v_{i,1},v_{i,2},\dots,v_{i,d},\dots,v_{i,D}]$, called donor vector, by the following equation:

$$\mathbf{v}_i=\mathbf{x}_{r_{1,i}}+F*(\mathbf{x}_{r_{2,i}}-\mathbf{x}_{r_{3,i}}), \tag{7}$$

  where F is a constant coefficient, $r_{1,i}$, $r_{2,i}$ and $r_{3,i}$ are three other random-selected individuals in the population.
- Crossover: Crossover generates a new solution $\mathbf{u}_i=[u_{i,1},u_{i,2},\dots,u_{i,d},\dots,u_{i,D}]$, called trail vector, via recombining $\mathbf{x}_i$ and $\mathbf{v}_i$ as follow:

$$u_{i,d}=\begin{cases} v_{i,d} & rnd_i=d \ or \ c_{i,d}\le CR \\ x_{i,d} & rnd_i\ne d \ and \ c_{i,d}>CR \end{cases}, \tag{8}$$

  where rnd_i is a random integer between 1 to D, $c_{i,d}$ is a random number between 0 to 1, CR is a constant coefficient representing the strength of varying. The existence of rnd_i guarantees that $\mathbf{u}_i$ differs from $\mathbf{x}_i$ at least in one dimension.
- Selection: If the fitness of $\mathbf{u}_i$ is better than $\mathbf{x}_i$, $\mathbf{x}_i$ will be replaced by $\mathbf{u}_i$. Otherwise $\mathbf{x}_i$ will be retained.

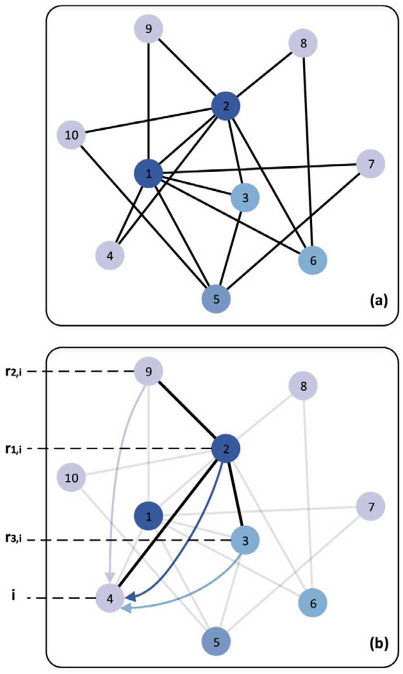

Fig. 4. Schematic diagram of networked differential evolution.

In DE, information flows among individuals via mutation operator. From the view of NITS, each individual utilizes information from three other individuals $r_{1,i}$, $r_{2,i}$ and $r_{3,i}$ to create a temp solution. Note that the three individuals are selected from the whole population, which implies that each individual could directly obtain information from anyone in the population. In other words, fully connected structure is adopted in traditional DE. Besides selecting all three information sources randomly, [7] also introduces other strategies, such as to deterministically select the best individual as $r_{1,i}$.

5.2. Networked differential evolution

To implement the NEA framework in DE, we introduce a variant named networked differential evolution (N-DE), which generalize DE in two aspects.

Firstly, we adopt a BA network as the population structure. As most individuals have few connections, the choices in mutation are limited. This limitation is particularly important to DE because individuals in DE naturally require three information sources, which is much larger than most EAs like GA or PSO.

Secondly, we modify the mutation operator for DE. When individual i is doing mutation operator, $r_{1,i}$ should be selected in i's neighbors, while $r_{2,i}$ and $r_{3,i}$ should directly connect with $r_{1,i}$. It should be mentioned that the donor vector is created based on $r_{1,i}$ and adding a difference term of $r_{2,i}$ and $r_{3,i}$ according to Eq. (8), which indicates that donor vector is generated around $r_{1,i}$. As the modification of N-DE restricts non-hub individuals to more probably choose hubs as $r_{1,i}$, the surrounding space of hub individuals will actually be exploited more frequently. Meanwhile, as hubs are more likely becoming $r_{1,i}$, it will provide a large set of neighbors to choose $r_{2,i}$ and $r_{3,i}$, further helping the exploitation of hubs.

Fig. 4 gives an example of N-DE with a population of 10 individuals. Black lines in Fig. 4 (a) indicates the BA network structure of the population. Fig. 4 (b) presents a scenario of the connection relationship when $i = 4$, where the activated connections are maintained black while others are lightened. In this scenario, individual 2 is selected as $r_{1,i}$ from 4's neighbors, then individual 3 and 9 are selected as $r_{2,i}$ and $r_{3,i}$ from $r_{1,i}$'s neighbors. The colored arrows imply the information flow from $r_{1,i}$, $r_{2,i}$ and $r_{3,i}$ to i.

5.3. Experiments

In the experiments of DE, 13 widely used benchmark problems are employed [7]. The details of these problems are shown in Table 6. Here, $f_1 \sim f_4$ are continuous unimodal functions. f_5 is Rosenbrock function which is unimodal when the dimension is less than 3, but it has multiple minima in higher dimensional cases. f_6 is a discontinuous step function, and f_7 is a noisy quartic function. $f_8 - f_{13}$ are multimodal and the number of their local minima increases exponentially with the problem dimension. Here we set $D = 30$ for the benchmark functions in the following experiments.

Table 6
Benchmark functions for differential evolution.

Formula	Range
$f_1(x)=\sum_{i=1}^{D} x_i^2$	$[-100,100]^D$
$f_2(x)=\sum_{i=1}^{D} \lvert x_i\rvert+\prod_{i=1}^{D} \lvert x_i\rvert$	$[-10,10]^D$
$f_3(x)=\sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2$	$[-100,100]^D$
$f_4(x)=\max_i\{\lvert x_i\rvert\}$	$[-100,100]^D$
$f_5(x)=\sum_{i=1}^{D-1} 100(x_{i+1}-x_i^2)^2+(x_i-1)^2$	$[-30,30]^D$
$f_6(x)=\sum_{i=1}^{D} (x_i+0.5)^2$	$[-100,100]^D$
$f_7(x)=\sum_{i=1}^{D} ix_i^4+random[0,1)$	$[-1.28,1.28]^D$
$f_8(x)=418.9829\times D+\sum_{i=1}^{D} x_i\sin(\lvert x_i\rvert^{\frac{1}{2}})$	$[-500,500]^D$
$f_9(x)=\sum_{i=1}^{D} x_i^2-10\cos 2\pi x_i+10$	$[-5.12,5.12]^D$
$f_{10}(x)=-20exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2})+20$	
$-exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} cos2\pi x_i})+e$	$[-32,32]^D$
$f_{11}(x)=\frac{1}{4000}\sum_{i=1}^{D} x_i^2-\prod_{i=1}^{D}\cos\frac{x_i}{\sqrt{i}}+1$	$[-600,600]^D$
$f_{12}(x)=\frac{\pi}{D}\{10\cdot\sin^2(\pi y_1)$	
$+\sum_{i=1}^{D-1} 0.125\cdot(x_i-1)^2\cdot[1+10\cdot\sin^2(\pi y_{i+1})]$	
$+0.125\cdot(x_D-1)^2\}+\sum_{i=1}^{D} u(x_i,10,100,4),$	
$y_i=1+\frac{1}{4}\cdot(x_i-1),$	
$u(z,a,k,m)=\begin{cases} k(z-a)^m & z>a \\ 0 & -a\le z\le a \\ k(-z-a)^m & z<-a \end{cases}$	$[-50,50]^D$
$f_{13}(x)=0.1\{\sin^2(3\pi x_1)$	
$+\sum_{i=1}^{D-1}(x_i-1)^2\cdot[1+\sin^2(3\pi x_{i+1})]$	
$+(x_D-1)^2\cdot[1+\sin^2(2\pi x_D)]+\sum_{i=1}^{D}$	
$+u(x_i,5,100,4),$	
$u(z,a,k,m)=\begin{cases} k(z-a)^m & z>a \\ 0 & -a\le z\le a \\ k(-z-a)^m & z<-a \end{cases}$	$[-50,50]^D$

Table 7
Results of average fitness value.

Functions	*Gen*	DE	N-DE
f_1	1500	$7.40E-21$	**3.26E-40**
f_2	2000	$4.94E-12$	**5.81E-25**
f_3	5000	$5.63E-64$	**4.63E-117**
f_4	5000	**1.01E-02**	$1.03E-02$
f_5	20000	**5.93E-30**	**5.93E-30**
f_6	1500	$2.71E-16$	**1.18E-31**
f_7	3000	$8.15E01$	**2.14E01**
f_8	9000	$8.29E01$	**5.92E01**
f_9	5000	$8.24E01$	**1.69E01**
f_{10}	2000	$2.27E-12$	**1.13E-15**
f_{11}	3000	**0.00E00**	**0.00E00**
f_{12}	1500	$5.03E-17$	**3.38E-17**
f_{13}	1500	$2.67E-16$	**2.88E-17**

The population size of DE and N-DE is set as $N=100$, and we set $F=0.5$, $CR=0.9$ and maintain other mechanisms following [7]. The parameters of BA network structure are set as $m_0=10$, $m=6$. The results of comparison are shown in Table 7, where each result is the average of 50 runs. The maximum iteration *Gen* are set following.

As shown in Table 7, N-DE outperforms DE on 10 problems, and shares the same performance with DE on f_5 and f_{11}. For f_4, only in which N-DE does not outperform DE, the performances of both algorithms are quite comparable. Note that N-DE achieves better results in all multimodal problems, showing the effectiveness of the NEA framework.

6. Conclusion

In this paper, we introduce the networked evolutionary algorithm framework for solving optimization problems. The NEA framework, based on the idea of NITS, improves EA by involving heterogeneity from aspects of the communication network structure and information fusion strategies. We further implement the NEA framework on three representative EAs: GA, PSO and DE. Despite the different details of implementation, all the results show the significant power of NEA on improving the performance of traditional EA.

Acknowledgment

This paper is supported by the National Key Research and Development Program of China (Grant no. 2016YFB1200100), the National Natural Science Foundation of China (Grant Nos. 61425014, 61521091, 91538204, 61671031 and 61722102), and by the Slovenian Research Agency (Grants Nos. J1-7009 and P5-0027).

References

[1] J.H. Holland, Adaptation in Nature and Artificial Systems, The MIT Press, Cambridge, MA, USA, 1992.
[2] X. Yao, Evolutionary Computation: Theory and Applications, World Scientific Publishing, Singapore, SG, 1999.
[3] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, Cambridge, UK, 2004.
[4] A.P. Engelbrecht, Computational Intelligence: An Introduction, 2nd ed., John Wiley, West Sussex, England, 2007.
[5] M. Mitchell, An Introduction to Genetic Algorithms, The MIT Press, Cambridge, MA, USA, 1996.
[6] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of the IEEE International Conference on Neural Networks, 1995, pp. 1942–1948.
[7] R. Stone, K. Price, Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces, J. Glob. Optim. 11 (1997) 341–359.
[8] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, J. Glob. Optim. 39 (2007) 459–471.
[9] G. Vrbančič, L. Brezočnik, U. Mlakar, D. Fister, I. Fister Jr., Niapy: Python microframework for building nature-inspired algorithms, J. Open Source Softw. 3 (2018) 613.
[10] Z.H. Zhan, J. Zhang, Y. Li, H.S. Chung, Adaptive particle swarm optimization, IEEE Trans. Syst. Man Cybern. Part B 39 (2009) 1362–1381.
[11] R. Mukherjee, G.R. Patra, R. Kundu, S. Das, Cluster-based differential evolution with crowding archive for niching in dynamic environments, Inf. Sci. 267 (2014) 58–82.
[12] B. Qu, P. Suganthan, S. Das, A distance-based locally informed particle swarm model for multimodal optimization, IEEE Trans. Evolut. Comput. 17 (2013) 387–402.
[13] X.D. Li, Niching without niching parameters: particle swarm optimization using a ring topology, IEEE Trans. Evolut. Comput. 14 (2010) 150–169.
[14] S.Z. Zhao, J.J. Liang, P.N. Suganthan, M.F. Tasgetiren, Dynamic multi-swarm particle swarm optimizer with local search for large scale global optimization, in: Proceedings of the IEEE Congress on Evolutionary Computation, 2008.
[15] J. Kennedy, R. Mendes, Population structure and particle swarm performance, in: Proceedings of the Congress on Evolutionary Computation, 2002, pp. 1671–1676.
[16] A. Ratnaweera, S.K. Halgamuge, H.C. Watson, Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients, IEEE Trans. Evolut. Comput. 8 (2004) 240–255.
[17] R. Mendes, J. Kennedy, J. Neves, The fully informed particle swarm: simpler, maybe better, IEEE Trans. Evolut. Comput. 8 (2004) 204–210.
[18] S. Segarra, W. Huang, A. Ribeiro, Diffusion and superposition distances for signals supported on networks, IEEE Trans. Sign. Inf. Process. Over Netw. 1 (2015) 20–32.
[19] X. Gong, X. Chen, J. Zhang, H.V. Poor, Exploiting social trust assisted reciprocity (STAR) toward utility-optimal socially-aware crowdsensing, IEEE Trans. Sign. Inf. Process. Over Netw. 1 (2015) 195–208.
[20] S. Shaghaghian, M. Coates, Optimal forwarding in opportunistic delay tolerant networks with meeting rate estimations, IEEE Trans. Sign. Inf. Process. Over Netw. 1 (2015) 104–116.
[21] A.L. Barabási, R. Albert, Emergence of scaling in random networks, Science 286 (1999) 509–512.
[22] C.H. Li, S.X. Yang, T.N. Trung, A self-learning particle swarm optimizer for global optimization problems, IEEE Trans. Syst. Man Cybern. Part B 42 (2012) 627–646.
[23] W.B. Du, Y. Gao, C. Liu, Z. Zheng, Z. Wang, Adequate is better: particle swarm optimization with limited-information, Appl. Math. Comput. 268 (2015) 832–838.
[24] R. Eberhart, Y.H. Shi, Particle swarm inspired evolutionary algorithm (PS-EA) for multi-criteria optimization problems, in: Proceedings of the Congress on Evolutionary Computation, 2003.
[25] X. Yao, Y. Liu, G.M. Lin, Evolutionary programming made faster, IEEE Trans. Evolut. Comput. 3 (1999) 82–102.
[26] J.J. Liang, A.K. Qin, P.N. Suganthan, S. Baska, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Trans. Evolut. Comput. 10 (2006) 281–295.
[27] I. Fister Jr., X.S. Yang, K. Ljubic, D. Fister, J. Brest, I. Fister, Towards the novel reasoning among particles in PSO by the use of RDF and SPARQL, Sci. World J. 2014 (2014) 121782.
[28] C. Liu, W.B. Du, W.X. Wang, Particle swarm optimization with scale-free interactions, PLoS ONE 9 (2014) e97822.
[29] Y.H. Shi, R. Eberhart, A modified particle swarm optimizer, in: Proceedings of the IEEE International Conference on Evolutionary Computation Proceedings, 1998, pp. 69–73.
[30] Y. Gao, W.B. Du, G. Yan, Selectively-informed particle swarm optimization, Sci. Rep. 5 (2015) 92–95.