

Received March 29, 2019, accepted April 21, 2019, date of publication April 25, 2019, date of current version May 7, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2913017

Prediction of Network Traffic of Smart Cities Based on DE-BP Neural Network

XIUQIN PAN, WANGSHENG ZHOU, YONG LU[✉], AND NA SUN

School of Information Engineering, Minzu University of China, Beijing 100081, China

Corresponding author: Yong Lu (yong@muc.edu.cn)

This work was supported in part by the Project of Young Teachers' Scientific Research Ability Promotion Program, and in part by the National Natural Science Foundation of China.

ABSTRACT Smart cities make full use of information technology so as to make intelligence responses to all requirements, including network and city services. This paper proposes a differential evolution back propagation (DE-BP) neural network traffic prediction model applicable for a smart cities network to predict the network traffic. The proposed approach takes the impact factor of network traffic as the input layer and the network traffic as the output layer and trains the DE-BP network with the past traffic data so as to obtain the mapping relationship between the impact factor and the network traffic and get the predicted value of the network traffic. The experimental results show that the proposed approach can accurately predict the trend of network traffic. Within the allowable error range, the predicted traffic volume is consistent with the actual traffic volume trend, and the predicted error is small.

INDEX TERMS Smart city, network traffic prediction, urban computing, BP neural network, global optimization.

I. INTRODUCTION

With the help of cloud computing, Internet of Things, device to device (D2D) communication, artificial intelligence and big data, urban computing and intelligence novel solutions can be created to improve urban environment, human life quality, and smart city systems [1], [2]. Smart cities enhance the construction of mobile telecommunication network, promote the applications of a new generation mobile telecommunication services, construct integrated service platform and satisfy the development requirements of new data services, new-type industry chain and business patterns; actively study and timely apply such new wireless communication techniques, develop the broadband wireless network with access diversification, network integration and integrated applications, further improve the existing network coverage and optimize the existing network, improve the user capacity, increase upstream and downstream bandwidth and build multi-layer and vertical infrastructure with high bandwidth and full coverage. Thus, urban computing has recently attracted significant attention from industry and academia for building smart cities.

The associate editor coordinating the review of this manuscript and approving it for publication was Rongbo Zhu.

BP network is a kind of multi-layer feed-forward neural network. Because of its simple structure, many adjustable parameters, training algorithms and good maneuverability, BP neural network has been extensively applied [3]. According to the statistics, 80% ~ 90% of the neural network models have adopted BP network or its transformations. BP network is the core part of forward network and it is the most essential and perfect part in the neural network. Although it is the most widely applied algorithm in the artificial neural network, BP neural network also has certain defects. For example, the learning and convergence speeds are too slow. It cannot guarantee to converge to the global minimum point. It is not easy to determine the network structure. Besides, the selection of network structure, the initial connection weight and threshold have a huge impact on the network training, but they cannot be obtained accurately. To overcome these defects, the neural network can be optimized with DE algorithm.

A general optimization problem aims to minimize or maximize a performance indicator by selecting a set of parameters. There exist many optimization problems, such as structure design, portfolio investment, economic dispatch, job scheduling, and water allocation. So, effective optimization methods are always required. Traditional optimization methods include gradient descent, conjugate gradient method, Newton method, momentum, Lagrange multiplier, and so on [4].

For the above algorithms, there are some advantages: perfect theory, small computation, fast convergence speed and definite termination criteria. But most of them strongly depend on initial values and only obtain local optimal solutions. For some complex and multimodal problems, traditional optimization methods encounter difficulties.

The rest of this paper is organized as follows. In Section 2, the related work is described. The novel differential evolution back propagation neural network traffic prediction model applicable for smart cities network is discussed in Section 3. In Section 4, experiment verifications and comparisons of DE algorithm are presented, and Section 5 concludes the paper with summary and future research directions.

II. RELATED WORKS

In the network management, the network performance monitoring and the traffic prediction function of smart cities are very important. From the research on network traffic, we can better understand the features of the backbone network of smart cities. With the applications of the tools and approaches of such new generation of information technology as the infrastructure of the Internet of Things (IoT), cloud computing and geographic space, big data and artificial intelligence, smart cities can achieve thorough perception, interconnection with extensive broadband, applications of intelligent combination and sustainable innovation featured by user innovation, opening innovation, mass innovation and collaborated innovation.

In order to timely and accurately provide the situation and information of the changes of future network traffic to guide and control such changes and prevent network blockage, network control, to a very large extent, depends on the prediction of network traffic. So, network traffic prediction is the foundation of network management. Along with the combined development of network and mobile technology as well as the innovation, smart cities under the context of knowledge is the advanced form of the development of informational city following digital city and it emphasizes comprehensive and sustainable development in economy, society and environment through value creation and people orientation [5], [6].

Recently, some intelligent optimization algorithms were proposed to solve different optimization problems [7], [8]. Differential evolution (DE) is one of the most popular evolutionary algorithms (EAs) proposed by Price and Storn [9]. The DE/rand strategies have strong exploration capacity, while the DE/best strategies are good at exploitation. Besides the classical mutation strategies, some improved versions were designed, such as neighborhood mutation [10], Gaussian sampling mutation [11], and triangular mutation [12]. There are two vital parameters including scale factor and crossover rate in DE. Although some studies have suggested some good choices to set those parameters, they are problem-dependent [13]. To improve this case, some adaptive parameter strategies were proposed. During the search process, the parameters are dynamically updated. Numerical experiments show these parameter strategies are effective.

Brest *et al.* [14] used random values to replace the parameters F and CR . These parameters are assigned to random values with the range $[0,1]$. Experiments showed that the modified DE outperforms classical DE and some other evolutionary algorithms. Tong *et al.* [15] used multi-population and ensemble techniques in DE. Results on CEC 2017 problems show the new method is superior to some other well-known DE variants. Fan *et al.* [16] designed a multi-algorithm method to automatically select suitable DE variants. Wu *et al.* [17] presented an improved ensemble of DE variants (EDEV), in which three popular DE variants were used. Awad *et al.* [18] used ensemble of parameters and niching based population reduction in a sinusoidal DE. A restart method is utilized to improve the quality of solution at later generation. To solving constrained optimization problems, Xu *et al.* [19] firstly introduced an adaptive method to generate trial solutions. Then, a cluster substitution method was utilized to generate feasible solutions. In [20], Zorarpacı and Özel presented a hybrid algorithm based on DE and ABC for feature selection. Kamboj *et al.* [21] proposed a novel DE with random search to solve multi-objective and multi-area unit commitment problem. Simulation results show that the proposed algorithm can achieve reasonable solutions.

BP neural network is constituted by two processes: the forward computing (forward propagation) of data flow and the back propagation of error signals [22], [23]. In the former process, the propagation direction is the input layer, the hidden layer and the output layer. The state of the neurons in each layer only affects the neurons in the next layer. If the expected output cannot be obtained in the output layer, the latter process will start [24], [25]. With the alternation of these two processes, it implements the gradient descent strategy of error function in the weight vector space and searches a group of weight vectors through dynamic iterations to minimize the error function of the network so as to complete information extraction and memorization. This paper designs a network traffic prediction model applicable for smart cities network to predict the network traffic. It is very necessary to launch some prediction work on network traffic as the evidence to improve the quality of network services [26], [27]. Fig. 1 shows the structure of BP neural network.

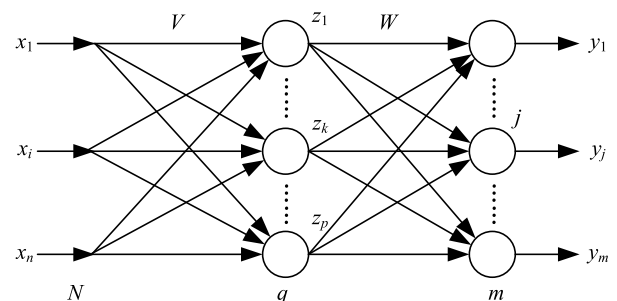


FIGURE 1. Structure of BP neural network.

Assume that there are n nodes in the input layer of BP neural network, q nodes in the hidden layer and m nodes in

the output layer and that the weight between the input layer and the hidden layer is v_{ki} and the weight between the hidden layer and the output layer is w_{jk} . The transfer function of the hidden layer is $f_1(g)$ and that of the output layer is $f_2(g)$. Then the output of the nodes in the hidden layer is (put the threshold into the sum term):

$$z_k = f_1\left(\sum_{i=0}^n v_{ki}x_i\right) \quad k = 1, 2, \dots, q \quad (1)$$

The output of the nodes in the output layer is as follows:

$$y_j = f_1\left(\sum_{k=0}^q w_{jk}z_k\right) \quad j = 1, 2, \dots, m \quad (2)$$

The back propagation of error is to calculate the output errors of the neurons in every layer starting from the output layer and then adjust the weights and thresholds of each layer based on error gradient descent method so as to make the final corrected output approximate the expected value [28], [29].

A. DEFINE THE ERROR FUNCTION

Input P learning samples and represent with $x^1, x^2, \dots, x^p, \dots, x^P$. Input the P th sample into the network and obtain the output of y_j^p ($j = 1, 2, \dots, m$). Use squared error function and obtain the error E_p of the P th sample.

$$E_p = \frac{1}{2} \sum_{j=1}^m (t_j^p - y_j^p)^2 \quad (3)$$

In which, y_j^p is the expected output.

B. THE CHANGES OF THE WEIGHT IN OUTPUT LAYER

Use accumulated error BP algorithm and adjust w_{jk} so as to minimize the global error E , namely

$$\Delta w_{jk} = -\eta \frac{\partial E}{\partial w_{jk}} = -\eta \frac{\partial}{\partial w_{jk}} \left(\sum_{p=1}^P E_p \right) = \sum_{p=1}^P \left(-\eta \frac{\partial E_p}{\partial w_{jk}} \right) \quad (4)$$

In which, η - is the learning rate.

The error signal is defined as:

$$\delta_{yj} = -\frac{\partial E_p}{\partial S_j} = -\frac{\partial E_p}{\partial y_j} g \frac{\partial y_j}{\partial S_j} \quad (5)$$

In this formula, the first term is

$$\frac{\partial E_p}{\partial y_j} = \frac{\partial}{\partial y_j} \left[\frac{1}{2} \sum_{j=1}^m (t_j^p - y_j^p)^2 \right] = -\sum_{j=1}^m (t_j^p - y_j^p) \quad (6)$$

The second term is:

$$\frac{\partial y_j}{\partial S_j} = f_2'(S_j) \quad (7)$$

is the partial differential of the transfer function in the output layer, then

$$\delta_{yj} = \sum_{j=1}^m (t_j^p - y_j^p) f_2'(S_j) \quad (8)$$

The following can be obtained through chain theorem:

$$\frac{\partial E_p}{\partial w_{jk}} = \frac{\partial E_p}{\partial S_j} g \frac{\partial S_j}{\partial w_{jk}} = -\delta_{yj} z_k = -\sum_{j=1}^m (t_j^p - y_j^p) f_2'(S_j) g z_k \quad (9)$$

The formula to adjust the weights of neurons in the output layer is as follows.

$$\Delta w_{jk} = \sum_{p=1}^P \sum_{j=1}^m \eta (t_j^p - y_j^p) f_2'(S_j) g z_k \quad (10)$$

III. PROPOSED APPROACH

In Section 4, we briefly introduce some representative mutation operators. Among these mutations, DE/rand and DE/best are widely used. For the first mutation, it shows very strong exploration ability, but it has poor exploitation capacity. For the second mutation, it does well in exploitation, but it has poor exploration capacity. Therefore, how to balance the exploration and exploitation abilities in the mutation scheme is a vital task.

In [13], Huang *et al.* combined a concept of best of random (BoR) with DE/rand/1. When conducting the BoR, three solutions X_{i1} , X_{i2} and X_{i3} are randomly selected. The above solutions are mutually different. At first, X_{i1} is compared with X_{i2} and X_{i3} , respectively. If X_{i2} is better than X_{i1} , we swap X_{i1} with X_{i2} . If X_{i3} is better than X_{i1} , we swap X_{i1} with X_{i3} . Thus, X_{i1} is the best solution among X_{i1} , X_{i2} and X_{i3} . The above comparison process can be formulated as below.

$$\text{swap}(X_{i1}, X_{i2}), \quad \text{if } f(X_{i2}) < f(X_{i1}) \quad (11)$$

$$\text{swap}(X_{i1}, X_{i3}), \quad \text{if } f(X_{i3}) < f(X_{i1}) \quad (12)$$

where $\text{swap}(X_{i1}, X_{i2})$ indicates X_{i1} and X_{i2} are exchanged, and $\text{swap}(X_{i1}, X_{i3})$ represents X_{i1} and X_{i3} are exchanged

For the BoR, it uses a local best solution X_{i1} as the first random individual in Eq. (11), and DE/best/1 employs the global best X_{best} in Eq. (12). The X_{i1} can improve the exploitation, but less than DE/best/1. So, the search characteristic of BoR is between the above two mutation schemes. Based on DE/BoR/1, we try to embed this concept into other mutations. In this work, we combine BoR and DE/rand/2 to construct a new mutation strategy DE/BoR/2. The detailed steps of DE/BoR/2 are listed in Algorithm 1. From Algorithm 1, X_{i1} is the best solution among five random chosen solutions. X_{i1} is a local best solution, which is worse than X_{best} with a high probability. The better the base vector (X_{i1} or X_{best}) is, the faster the convergence is. However, the fast convergence speed may lead to premature convergence. Because new solutions are near to the first random individual according to the mutation scheme. If all solutions are near to the same individual, the diversity of population will decrease. The DE/BoR/2 can prevent this case. The local best solution X_{i1} can guide the search and X_{i1} is dynamically updated with the changing of X_{i1} . The procedure of our approach DE/BoR/2 is listed in Algorithm 1.

Algorithm 1 The Proposed DE/BoR/2

```

Begin
  Randomly initialize the population at generate  $t = 0$ ;
  while the termination rule is not met do
    for  $i = 1$  to  $N$  do
      Create the  $V_i$  according to Algorithm 1;
      Create the  $U_i$  in terms of Eq. (15);
    end for
    Execute the selection in terms of Eq. (16);
    end while
    Update the global best solution  $X_{best}$ ;
  end while
End
  
```

To optimize BP neural network with DE is to optimize the initial weight and threshold of BP neural network with DE algorithm so that the optimized BP neural network can better predict the samples. The elements to optimize BP neural network with DE algorithm include: the initialization of the population, the fitness function, the selection operator, the cross-over operator and the mutation operator. After the optimization of DE algorithm, get the best initial weight and threshold matrix, substitute that initial weight and threshold into the network and draw the training error value, the prediction value, the predicted error and the training error and so on [30], [31]. The procedure to optimize BP neural network with DE algorithm is shown as Fig. 2.

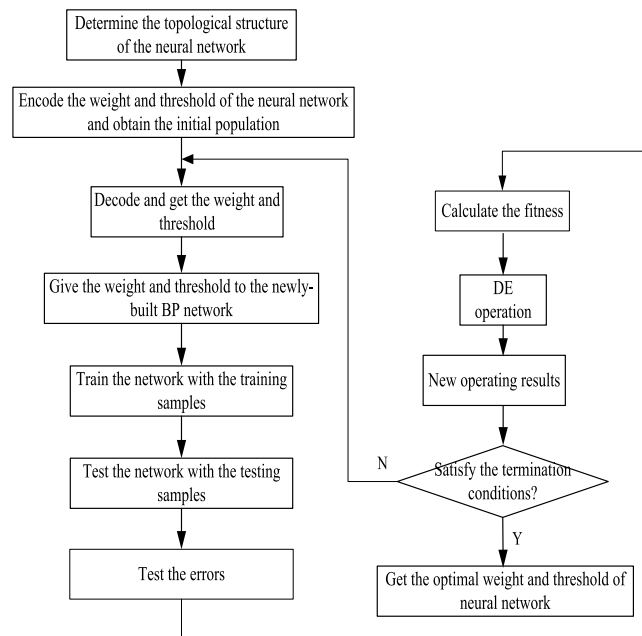


FIGURE 2. The procedure to optimize BP neural network with DE.

To optimize BP neural network with DE includes the determination of BP neural network structure, the optimization of weight and threshold with DE algorithm as well as the training and prediction of BP neural network [32], [33]. Among them, the topological structure of BP neural network is

determined according to the number of input/output parameters of the samples [34], [35]. In this way, the number of parameters to be optimized with DE algorithm can be optimized so as to determine the coding length of the population individuals. As it is the initial weight and threshold of BP neural network that are optimized with DE algorithm, the number of weights and thresholds can be obtained as long as the network structure is known [36], [37]. The weights and thresholds of the neural network are generally the random numbers within the scope of $[-0.5, 0.5]$ which are randomly initialized [38]. These initialized parameters play a great impact on the network training, but it cannot be accurately obtained. For the same initial weight and threshold, the training results of the network are the same. The introduction of DE algorithm is to optimize the optimal initial weight and threshold.

The purpose of optimizing BP neural network with DE algorithm is to obtain the initial weight and threshold of the network through DE algorithm and its basic idea is to represent the initial weight and threshold of the network with individuals and the predicted error of BP neural network the individual value of which is initialized will be taken as the fitness value of that individual and then find the initial weight of the optimal BP neural network by searching the best individual.

IV. SIMULATION EXPERIMENT

A. TEST FUNCTIONS USED IN THE EXPERIMENTS

In the numerical experiments, the proposed DE/BoR/2 is tested on some famous test functions. There are thirteen test functions, which were used in many optimization references [39]–[41]. In Table 1, the detailed test functions are presented, where D is the dimensional size.

B. PARAMETER SENSITIVITY ANALYSIS IN DE/BoR/2

In DE/BoR/2, there are two vital parameters and are used in DE/rand/1 [7]. When the mutation strategy is changed, the corresponding parameters may need to be adjusted. Therefore, we investigate different sets of and in DE/BoR/2 and select the best parameter setting. In this section, and are tested on different combinations. As seen, there are 16 different parameter settings. In the following experiments, DE/BoR/2 is run on 16 parameter combinations, respectively, and the best parameter combination is chosen among the comparisons. For other parameters, the maximum number of fitness values (MAXFEs) is equal to $2.0E + 05$ and N is set to 100.

Table 2 gives the mean fitness values of DE/BoR/2 when $CR = 0.1$. We can see that a small F is good for unimodal functions. For functions $f1$ - $f3$, $f5$, and $f7$, the results become worse with increasing of F . For multimodal functions, the value of F is hard to choose. For $f8$, a large F is better, while a small F is better for $f9$, $f10$, and $f12$. For $f11$ and $f13$, F between 0.25 and 0.5 is a good choice.

TABLE 1. Benchmark functions used in the experiments.

FUNCTIONS	SEARCH RANGE	D	GLOBAL OPTIMUM
$f_1(x) = \sum_{i=1}^D x_i^2$	[-100,100]	30	0
$f_2(x) = \sum_{i=1}^D x_i + \prod_{i=1}^D x_i$	[-10,10]	30	0
$f_3(x) = \sum_{i=1}^D (\sum_{j=1}^i x_j)^2$	[-100,100]	30	0
$f_4(x) = \max_i (x_i , 1 \leq i \leq D)$	[-100,100]	30	0
$f_5(x) = \sum_{i=1}^D [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	[-30,30]	30	0
$f_6(x) = \sum_{i=1}^D (\lfloor x_i + 0.5 \rfloor)^2$	[-100,100]	30	0
$f_7(x) = \sum_{i=1}^D i x_i^4 + rand[0,1)$	[-1.28,1.28]	30	0
$f_8(x) = \sum_{i=1}^D -x_i \sin(\sqrt{ x_i })$	[-500,500]	30	-12569.5
$f_9(x) = \sum_{i=1}^D [x_i^2 - 10 \cos(2\pi x_i) + 10]$	[-5.12, 5.12]	30	0
$f_{10}(x) = -20 \cdot \exp\left(-0.2 \cdot \sqrt{\frac{1}{D} \sum_{i=1}^D x_i^2}\right) - \exp\left(\frac{1}{D} \sum_{i=1}^D \cos(2\pi x_i)\right) + 20 + e$	[-32,32]	30	0
$f_{11}(x) = \frac{1}{4000} \sum_{i=1}^D x_i^2 - \prod_{i=1}^D \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	[-600,600]	30	0
$f_{12}(x) = 0.1 \left[\sin^2(3\pi x_1) + \sum_{i=1}^{D-1} (x_i - 1)^2 [1 + \sin^2(3\pi x_{i+1})] + (x_D - 1)^2 [1 + \sin^2(2\pi x_D)] \right] + \sum_{i=1}^D u(x_i, 10, 100, 4)$	[-50,50]	30	0
$f_{13}(x) = \frac{\pi}{D} \left[10 \sin^2(3\pi y_1) + \sum_{i=1}^{D-1} (y_i - 1)^2 [1 + \sin^2(3\pi y_{i+1})] + (y_D - 1)^2 [1 + \sin^2(2\pi y_D)] \right] + \sum_{i=1}^D u(y_i, 5, 100, 4)$	[-50,50]	30	0

TABLE 2. Mean fitness values of DE/BoR/2 when CR = 0.1.

Functions	F=0.1	F=0.25	F=0.5	F=0.7
	CR=0.1	CR=0.1	CR=0.1	CR=0.1
f1	3.58E-59	5.16E-45	2.08E-20	9.05E-10
f2	6.90E-36	3.14E-25	7.97E-12	3.40E-06
f3	7.99E+02	2.07E+03	6.52E+03	5.83E+03
f4	2.15E+00	2.34E-03	5.51E-01	5.95E+00
f5	6.20E+00	2.68E+01	2.73E+01	5.64E+01
f6	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f7	4.79E-03	7.56E-03	1.97E-02	4.93E-02
f8	-3.39E+03	-8.68E+03	-1.26E+04	-1.26E+04
f9	0.00E+00	0.00E+00	5.39E-02	3.35E+01
f10	7.69E-15	1.12E-14	3.67E-10	4.21E-05
f11	9.99E-16	0.00E+00	0.00E+00	4.27E-09
f12	3.46E-31	1.57E-32	3.58E-21	9.41E-11
f13	5.64E-08	1.35E-32	4.27E-21	4.30E-10

Table 3 lists the mean fitness values of DE/BoR/2 when CR = 0.3. From the results, F = 0.25 achieves better solutions than other cases on many functions. On f1, f2, f4, f10, f11, and f12, DE/BoR/2 with F = 0.25 obtains much better solutions than other F values. However, for f3 and f8, other F values can achieves better solutions.

TABLE 3. Mean fitness values of DE/BoR/2 when CR = 0.3.

Functions	F=0.1	F=0.25	F=0.5	F=0.7
	CR=0.3	CR=0.3	CR=0.3	CR=0.3
f1	4.45E+00	9.91E-73	2.66E-16	3.73E-02
f2	3.66E-25	7.22E-41	6.93E-10	2.57E-02
f3	1.31E+01	3.12E+02	1.12E+04	1.32E+04
f4	4.74E+00	1.31E-09	6.15E-01	1.58E+01
f5	1.42E+03	2.21E+01	2.49E+01	4.94E+02
f6	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f7	1.03E-02	3.34E-03	1.37E-02	6.45E-02
f8	-3.09E+03	-1.11E+04	-1.26E+04	-1.26E+04
f9	2.99E+00	5.67E+01	8.44E+01	1.16E+02
f10	3.55E-02	4.14E-15	8.65E-09	7.14E-02
f11	2.60E-05	0.00E+00	0.00E+00	3.39E-01
f12	2.43E-01	1.57E-32	4.28E-16	4.84E-01
f13	3.77E+00	1.35E-32	2.31E-16	2.75E-01

TABLE 4. Mean fitness values of DE/BoR/2 when CR = 0.55.

Functions	F=0.1	F=0.25	F=0.5	F=0.7
	CR=0.55	CR=0.55	CR=0.55	CR=0.55
f1	1.29E+02	7.52E-102	1.00E-08	5.60E+02
f2	1.71E-02	1.38E-55	2.16E-05	9.60E+00
f3	4.03E+02	6.79E-01	1.59E+04	1.58E+04
f4	1.95E+01	6.36E-07	4.21E+00	3.25E+01
f5	5.13E+04	5.53E+00	2.45E+01	2.36E+05
f6	2.90E+01	0.00E+00	0.00E+00	6.09E+02
f7	1.20E+00	2.13E-03	2.15E-02	3.83E-01
f8	-3.05E+03	-1.16E+04	-8.92E+03	-9.00E+03
f9	6.36E+00	9.97E+01	1.58E+02	1.77E+02
f10	2.60E+00	4.14E-15	4.50E-05	6.47E+00
f11	2.20E-01	0.00E+00	1.26E-08	1.54E+00
f12	3.79E+01	1.57E-32	2.45E-06	1.26E+04
f13	6.54E+01	1.35E-32	6.78E-07	6.10E+05

Table 4 shows the mean fitness values of DE/BoR/2 when CR = 0.55. It is obvious that F = 0.25 outperforms other F values on many functions. On f1, f2, f4, f6, and f10- f13, F = 0.25 can help DE/BoR/2 find reasonable solutions, but other F values falls into local optimum. For f9, F = 0.1 is much better than F = 0.25.

Table 5 displays the mean fitness values of DE/BoR/2 when CR = 0.9. From the results, DE/BoR/2 with different F cannot achieve promising solutions. For the majority of test functions, DE/BoR/2 is difficult to find reasonable solutions. So, CR = 0.9 is not a good choice among all parameter combinations.

From the above results, there are 16 cases for setting the parameters F and CR. To select the best case among 16 parameter combinations, Friedman statistical test is used to rank the performance of all cases. Table 6 gives the mean rank values. A smaller rank value means that it has a better rank. It can be seen that F = 0.25, CR = 0.55 achieves the best rank among 16 parameter combinations. It means that F = 0.25, CR = 0.55 obtains the best performance on the benchmark set. In addition, F = 0.25, CR = 0.3 is also a good choice, and it obtains the second place according to the rank

TABLE 5. Mean fitness values of DE/BoR/2 when CR = 0.9.

Functions	F=0.1	F=0.25	F=0.5	F=0.7
	CR=0.9	CR=0.9	CR=0.9	CR=0.9
f1	2.97E+03	2.52E+01	7.08E-04	1.22E+04
f2	2.64E+01	1.77E-02	1.08E-01	4.26E+01
f3	7.65E+03	4.95E+01	7.93E+02	2.17E+04
f4	3.75E+01	1.13E+01	5.52E+00	4.84E+01
f5	7.30E+05	1.69E+02	2.57E+01	8.85E+06
f6	3.13E+03	4.00E+00	0.00E+00	1.19E+04
f7	3.53E+00	6.80E-03	3.38E-02	3.67E+00
f8	-2.22E+03	-5.22E+03	-6.02E+03	-6.93E+03
f9	7.20E+01	8.96E+00	1.91E+02	2.31E+02
f10	9.52E+00	2.93E-03	9.97E-03	1.59E+01
f11	3.00E+00	1.62E-01	7.11E-05	1.59E+01
f12	4.12E+06	1.26E-01	7.58E-02	3.40E+06
f13	7.25E+06	3.86E+00	5.03E-02	2.73E+07

TABLE 6. Mean rank values.

DE/BoR/2	Mean rank
F=0.1, CR=0.1	4.96
F=0.25, CR=0.1	4.50
F=0.5, CR=0.1	5.31
F=0.7, CR=0.1	7.38
F=0.1, CR=0.3	8.46
F=0.25, CR=0.3	3.27
F=0.5, CR=0.3	6.15
F=0.7, CR=0.3	10.69
F=0.1, CR=0.55	11.69
F=0.25, CR=0.55	2.88
F=0.5, CR=0.55	8.54
F=0.7, CR=0.55	13.38
F=0.1, CR=0.9	14.38
F=0.25, CR=0.9	9.46
F=0.5, CR=0.9	9.46
F=0.7, CR=0.9	15.46

values. Based on above analysis, F = 0.25, CR = 0.55 is used in DE/BoR/2.

C. COMPARISON BETWEEN DE/BoR/2 AND CLASSICAL DE

In the following, we compare DE/BoR/2 with other classical DEs containing DE/rand/1, DE/rand/2 and DE/best/1. In the comparisons, all algorithms use the same termination rule and population size. Like Section 7.2, MAXFEs and N use the same values. For DE/BoR/2, F = 0.25 and CR = 0.55 are used according to the analysis of Section 7.2. DE/rand/2 and DE/BoR/2 employ the same parameters.

Table 7 displays the mean best fitness values of four DE algorithms. From Table 7, DE/best/1 outperforms other three DEs on f1, f2, f3, and f5. Especially for f5, only DE/best/1 achieves promising solutions. However, it falls into local minima on f6 and f11, but other DE algorithms converge to the global optimum. DE/rand/2 is better other DEs on f8, and it is slightly better than DE/BoR/2. For f9, all DEs are trapped into local optima, but DE/BoR/2 finds better solutions than

TABLE 7. Comparison between DE/BoR/2 and classical DEs.

Functions	DE/rand/1	DE/best/1	DE/rand/2	DE/BoR/2
	Mean	Mean	Mean	Mean
f1	3.68E-23	6.88E-245	1.36E-49	7.52E-102
f2	1.64E-11	2.39E-67	2.99E-28	1.38E-55
f3	1.17E-02	5.34E-54	1.64E+03	6.79E-01
f4	7.52E-05	1.36E-06	1.58E-06	6.36E-07
f5	7.78E+00	1.91E-26	2.66E+01	5.53E+00
f6	0.00E+00	6.60E+01	0.00E+00	0.00E+00
f7	5.86E-03	2.26E-02	6.30E-03	2.13E-03
f8	-7.76E+03	-4.00E+03	-1.19E+04	-1.16E+04
f9	1.76E+02	1.17E+02	1.03E+02	9.97E+01
f10	3.69E-12	7.16E+00	4.14E-15	4.14E-15
f11	0.00E+00	4.66E-02	0.00E+00	0.00E+00
f12	1.98E-23	5.18E-01	1.57E-32	1.57E-32
f13	2.01E-23	1.32E+00	1.35E-32	1.35E-32

other DEs. For f10-f13, DE/BoR/2 and DE/rand/2 are almost the same.

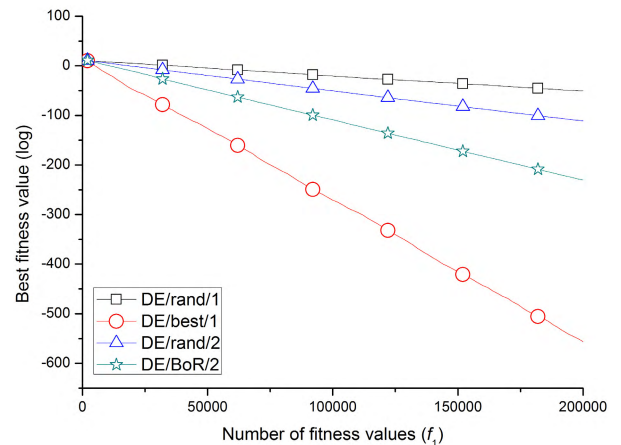


FIGURE 3. The search curves of four DEs on f1.

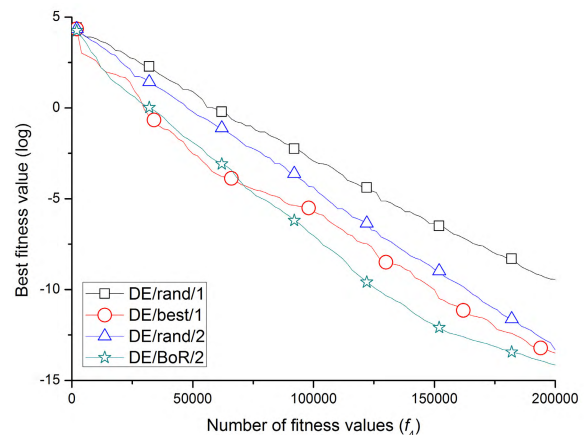


FIGURE 4. The search curves of four DEs on f4.

Figs. 3-7 show the search curves of four DEs on five representative functions. On function f1, DE/best/1 converges

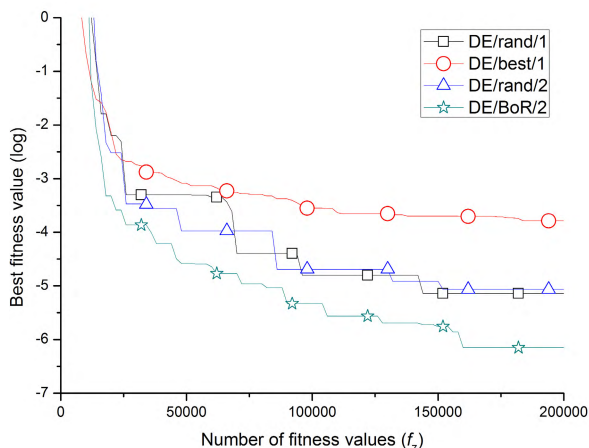


FIGURE 5. The search curves of four DEs on f_7 .

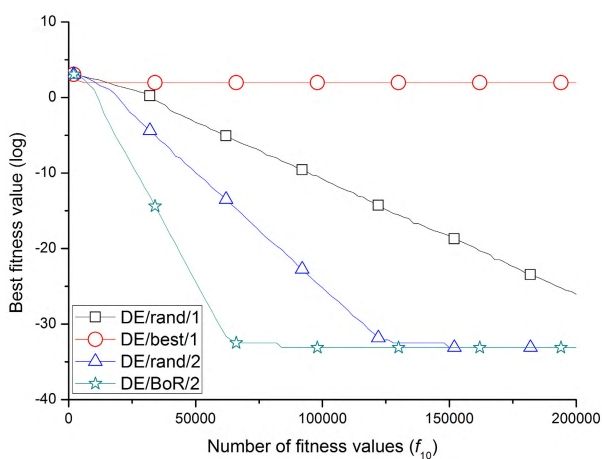


FIGURE 6. The search curves of four DEs on f_{10} .

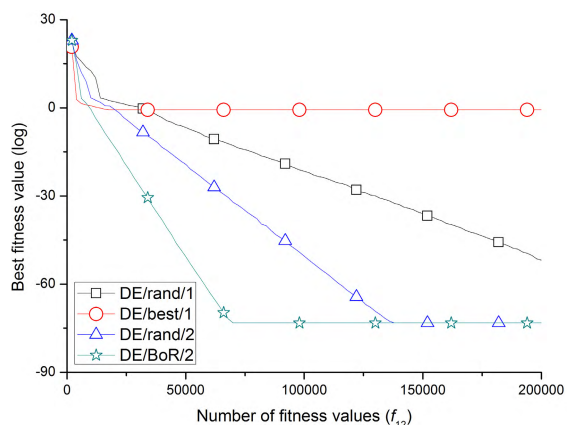


FIGURE 7. The search curves of four DEs on f_{12} .

faster than other three DEs and DE/BoR/2 is faster than two DE/rand algorithms. For function f_4 , DE/BoR is the fastest algorithm among four DEs. For DE/best/1, it is noted that its convergence speed outperforms DE/BoR/2 at the middle search stage. For function f_7 , DE/BoR and

TABLE 8. Mean rank values.

Algorithms	Mean rank
DE/rand/1	3.00
DE/best/1	2.85
DE/rand/2	2.42
DE/BoR/2	1.73

DE/best/1 are the fastest and slowest algorithms, respectively. The search curves are similar for functions f_{10} and f_{12} . Though DE/BoR/2 and DE/rand/2 achieve the same solutions, DE/BoR/2 is faster than DE/rand/2.

The mean rank values of the four DEs on thirteen test functions are computed based on Friedman test. Table 8 displays the mean rank of four DEs. Results show that DE/BoR/2 obtains the best rank, and it is better than the other three DEs.

Based on Matlab 2014a software platform, this paper uses the traffic data of backbone network collected and saved in the database during a certain time period as well as the time data of the corresponding period. Comparing the data predicted by the algorithm with the actual traffic data, using the original traffic data as sample data training, the network traffic in the next 120 hours is predicted. The maximum traffic is 301.1873 GB/hour and the minimum traffic is 11.6340 GB/hour. The difference between the two is very large, which shows that the network traffic is indeed an unstable time series. Trains and predicts with DE-BP neural network model in this paper and sets the termination conditions of network training as follows: the fitness value is not higher than 0.02, the initial neurons in the hidden layer is 30 and the maximum number of iterations of 200. The transfer function of input layer and hidden layer is Sigmoid type, and the output layer is linear transfer function. Because the range of Sigmoid function is $[0,1]$, in order to improve the convergence speed of the network, the samples are normalized and transformed to $[0,1]$. The training results of DE-BP neural network are shown in Figure 8, and the compared results of the actual and predicted network traffic are shown in Figure 9, in which the horizontal axis is the time period and the vertical axis is the network traffic.

According to the same parameters set above, the training results of the conventional BP neural network are shown in Fig. 10, and the simulated results of the actual and predicted network traffic are shown in Fig. 11.

From the above figures, it can be seen that the predicted flow of DE-BP neural network in the next 120 hours reflects the actual flow better in the trend of change, the speed of change and the degree of dispersion, which also verifies the validity and accuracy of the DE-BP neural network proposed in this paper.

After the training and validation of DE-BP and BP neural network, the performance of the two methods is analyzed and compared by means of average absolute percentage error and root mean square error. The analyzed results are compared in Table 9.

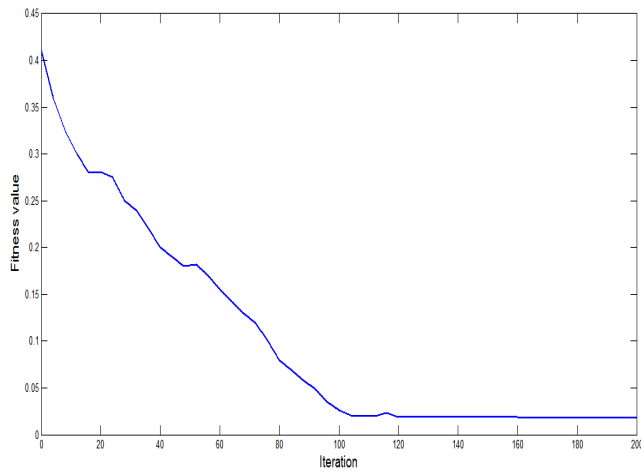


FIGURE 8. Training results diagram of DE-BP neural network.

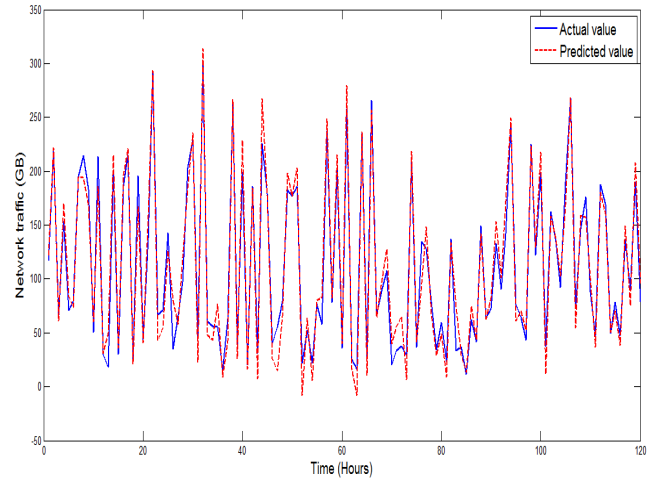


FIGURE 11. The predicted results of conventional BP neural network.

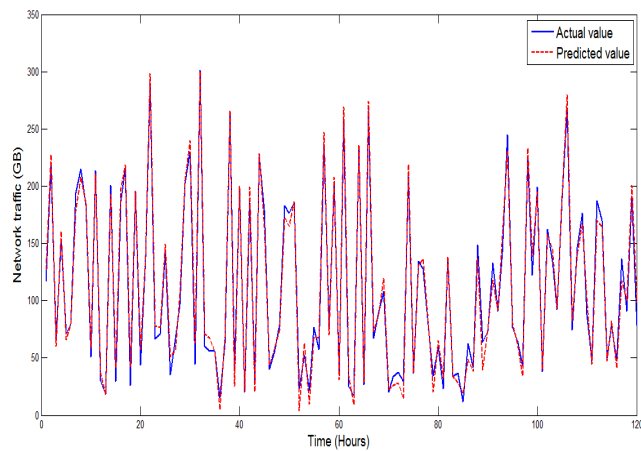


FIGURE 9. The predicted results of DE-BP neural network.

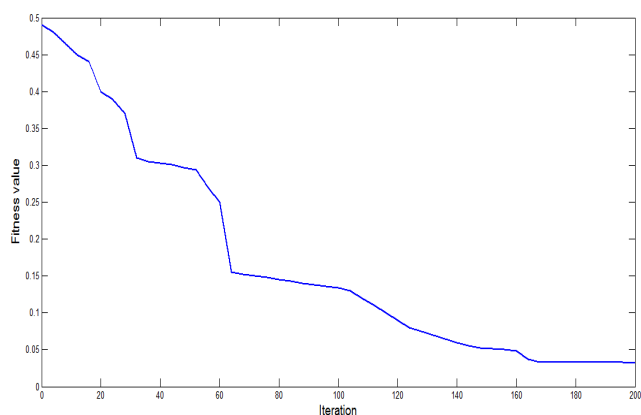


FIGURE 10. Training results diagram of conventional BP neural network.

Through the above experimental analysis, the proposed approach can accurately predict the trend of network traffic. Within the allowable error range, the predicted traffic volume is consistent with the actual traffic volume trend, and the predicted error is small. It improves the shortcomings

TABLE 9. The performance comparison of two methods.

Method	Number of hidden layer nodes	Average absolute percentage error	Root mean square error
DE-BP neural network	30	0.003413	0.002262
Conventional BP neural network	30	0.004327	0.003586

of local minimum and slow convergence speed of network traffic prediction. DE-BP neural network is an important tool for dealing with large-scale data problems because of its intelligent processing functions such as learning, memory and computation. Because the non-linear characteristics of network traffic can be depicted, the traffic prediction method based on neural network can show better performance than the conventional linear prediction method. At the same time, because the neural network has the “non-linear change law” of the memory signal in the training process, the prediction step has little influence on the prediction results, so it is suitable for long-term prediction. Network traffic varies with location and time. Because of the complexity, variability and heterogeneity of smart city network environment, the new network technology also makes the internet environment more complex and has new impacts on the characteristics of network traffic, therefore, traffic prediction needs to be adjusted constantly.

The algorithm in this paper is based on the main characteristics of network traffic, can improve the prediction accuracy of the neural network, accurately predict the change trend of network traffic. The trend of predicted traffic is generally the same as that of actual traffic, within the allowable error range, the prediction error is relatively low. With the rapid expansion of network scale in smart cities and the large-scale various service application based on network, the network

traffic prediction model proposed in this paper can effectively and accurately describe the characteristics of network traffic, and manages of network performance, service quality and access control.

V. CONCLUSION

This paper mainly study that the prediction of network traffic of smart cities based on DE-BP neural network. The key to construct DE-BP neural network is to seek the value of the three connection weights between the layers with training. Make full use of the global search of DE, find the best group of solutions through selection, cross-over and mutation and thus the DE-BP neural network is built. Then, take the impact factor of network traffic as the input layer and the network traffic as the output layer and train the DE-BP network with the past traffic data so as to obtain the mapping relationship between the impact factor and the network traffic and get the predicted value of the network traffic. How to effectively adjust the parameters of DE algorithm in order to improve the effectiveness of BP neural network is a further work in the future research.

CONFLICT OF INTEREST

We declare that there are no commercial or associative interest that represents a conflict of interest in connection with the work submitted.

REFERENCES

- [1] M. Dorigo, V. Maniezzo, and A. Colomi, "Ant system: Optimization by a colony of cooperating agents," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 26, no. 1, pp. 29–41, Feb. 1996.
- [2] W. Gong, A. Zhou, and Z. Cai, "A multioperator search strategy based on cheap surrogate models for evolutionary optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 5, pp. 746–758, Oct. 2015.
- [3] G. Wu, R. Mallipeddi, and P. N. Suganthan, "Ensemble strategies for population-based optimization algorithms—A survey," *Swarm Evol. Comput.*, vol. 44, pp. 695–711, Feb. 2019. doi: 10.1016/j.swevo.2018.08.015.
- [4] F. Wang, H. Zhang, Y. Li, Y. Zhao, and Q. Rao, "External archive matching strategy for MOEA/D," *Soft Comput.*, vol. 22, no. 23, pp. 7833–7846, Dec. 2018.
- [5] H. Wang, W. Wang, Z. Cui, X. Zhou, J. Zhao, and Y. Li, "A new dynamic firefly algorithm for demand estimation of water resources," *Inf. Sci.*, vol. 438, pp. 95–106, Apr. 2018.
- [6] M. Zhang, H. Wang, Z. Cui, and J. Chen, "Hybrid multi-objective cuckoo search with dynamical local search," *Memetic Comput.*, vol. 10, no. 4, pp. 199–208, Jul. 2018.
- [7] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, 1997.
- [8] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, Feb. 2011.
- [9] S. Das, S. S. Mullick, and P. N. Suganthan, "Recent advances in differential evolution—An updated survey," *Swarm Evol. Comput.*, vol. 27, pp. 1–30, Apr. 2016.
- [10] S. Das, A. Abraham, U. K. Chakraborty, and A. Konar, "Differential evolution using a neighborhood-based mutation operator," *IEEE Trans. Evol. Comput.*, vol. 13, no. 3, pp. 526–553, Jun. 2009.
- [11] B. Wu, X. Yan, Y. Wang, and C. G. Soares, "An evidential reasoning-based cream to human reliability analysis in maritime accident process," *Risk Anal.*, vol. 37, no. 10, pp. 1936–1957, Oct. 2017.
- [12] A. W. Mohamed, "An improved differential evolution algorithm with triangular mutation for global numerical optimization," *Comput. Ind. Eng.*, vol. 85, pp. 359–375, Jul. 2015.
- [13] Z. Huang, G. Shan, J. Cheng, and J. Sun, "TRec: An efficient recommendation system for hunting passengers with deep neural networks," *Neural Comput. Appl.*, vol. 31, pp. 209–222, Jan. 2019. doi: 10.1007/s00521-018-3728-2.
- [14] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Žumer, "Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems," *IEEE Trans. Evol. Comput.*, vol. 10, no. 6, pp. 646–657, Dec. 2006.
- [15] L. Tong, M. Dong, and C. Jing, "An improved multi-population ensemble differential evolution," *Neurocomputing*, vol. 290, pp. 130–147, May 2018.
- [16] Q. Fan, X. Yan, and Y. Zhang, "Auto-selection mechanism of differential evolution algorithm variants and its application," *Eur. J. Oper. Res.*, vol. 270, no. 2, pp. 636–653, Oct. 2018.
- [17] G. Wu, X. Shen, H. Li, H. Chen, A. Lin, and P. N. Suganthan, "Ensemble of differential evolution variants," *Inf. Sci.*, vol. 423, pp. 172–186, Jan. 2018.
- [18] N. H. Awad, M. Z. Ali, and P. N. Suganthan, "Ensemble of parameters in a sinusoidal differential evolution with niching-based population reduction," *Swarm Evol. Comput.*, vol. 39, pp. 141–156, Apr. 2018.
- [19] B. Xu, X. Chen, and L. Tao, "Differential evolution with adaptive trial vector generation strategy and cluster-replacement-based feasibility rule for constrained optimization," *Inf. Sci.*, vol. 435, pp. 240–262, Apr. 2018.
- [20] E. Zorarpacı and S. A. Özel, "A hybrid approach of differential evolution and artificial bee colony for feature selection," *Expert Syst. Appl.*, vol. 62, pp. 91–103, Nov. 2016.
- [21] V. K. Kamboj, S. K. Bath, and J. S. Dhillon, "Multiobjective multiarea unit commitment using hybrid differential evolution algorithm considering import/export and tie-line constraints," *Neural Comput. Appl.*, vol. 28, no. 11, pp. 3521–3536, Nov. 2017.
- [22] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398–417, Apr. 2009.
- [23] Y. Wang, E. Zio, X. Wei, D. Zhang, and B. Wu, "A resilience perspective on water transport systems: The case of eastern star," *Int. J. Disaster Risk Reduction*, vol. 33, pp. 343–354, Feb. 2019.
- [24] H. Wang, Z. Wu, and S. Rahnamayan, "Enhanced opposition-based differential evolution for solving high-dimensional continuous optimization problems," *Soft Comput.*, vol. 15, no. 11, pp. 2127–2140, Nov. 2011.
- [25] J. Zhang and A. C. Sanderson, "JADE: Adaptive differential evolution with optional external archive," *IEEE Trans. Evol. Comput.*, vol. 13, no. 5, pp. 945–958, Oct. 2009.
- [26] K. Opara and J. Arabas, "Comparison of mutation strategies in Differential Evolution—A probabilistic perspective," *Swarm Evol. Comput.*, vol. 39, pp. 53–69, Apr. 2018.
- [27] X. He and Y. Zhou, "Enhancing the performance of differential evolution with covariance matrix self-adaptation," *Appl. Soft Comput.*, vol. 64, pp. 227–243, Mar. 2018.
- [28] X. Zhang, H.-D. Chiang, and W. Chen, "TRUST-TECH-enhanced differential evolution methodology for box-constrained nonlinear optimisation," *Int. J. Bio-Inspired Comput.*, vol. 10, no. 1, pp. 1–11, 2017.
- [29] Y. Cai, J. Liao, T. Wang, Y. Chen, and H. Tian, "Social learning differential evolution," *Inf. Sci.*, vols. 433–434, pp. 464–509, Mar. 2018.
- [30] L. Skanderova, T. Fabian, and I. Zelinka, "Differential evolution based on node strength," *Int. J. Bio-Inspired Comput.*, vol. 11, no. 1, pp. 34–45, 2018.
- [31] W. S. Sakr, R. A. EL-Sehiemy, and A. M. Azmy, "Adaptive differential evolution algorithm for efficient reactive power management," *Appl. Soft Comput.*, vol. 53, pp. 336–351, Jun. 2017.
- [32] J. Kumar and A. K. Singh, "Workload prediction in cloud using artificial neural network and adaptive differential evolution," *Future Gener. Comput. Syst.*, vol. 81, pp. 41–52, Apr. 2018.
- [33] H. Zhu, Y. C. He, X. Z. Wang, and E. C. C. Tsang, "Discrete differential evolutions for the discounted 0-1 knapsack problem," *Int. J. Bio-Inspired Comput.*, vol. 10, no. 4, pp. 219–238, Jan. 2017.
- [34] Z. Sun, N. Wang, Y. Bi, and D. Srinivasan, "Parameter identification of PEMFC model based on hybrid adaptive differential evolution algorithm," *Energy*, vol. 90, pp. 1334–1341, Oct. 2015.
- [35] R. K. Singla and R. Das, "A differential evolution algorithm for maximizing heat dissipation in stepped fins," *Neural Comput. Appl.*, vol. 30, no. 10, pp. 3081–3093, Nov. 2018.
- [36] B. H. Zhou, L. M. Hu, and Z. Y. Zhong, "A hybrid differential evolution algorithm with estimation of distribution algorithm for reentrant hybrid flow shop scheduling problem," *Neural Comput. Appl.*, vol. 30, no. 1, pp. 193–209, Jul. 2018.
- [37] V. Ho-Huu, T. Nguyen-Thoi, T. Truong-Khac, L. Le-Anh, and T. Vo-Duy, "An improved differential evolution based on roulette wheel selection for shape and size optimization of truss structures with frequency constraints," *Neural Comput. Appl.*, vol. 29, no. 1, pp. 167–185, Jan. 2018.

- [38] R. P. Parouha, K. N. Das, "Economic load dispatch using memory based differential evolution," *Int. J. Bio-Inspired Comput.*, vol. 11, no. 3, pp. 159–170, 2018.
- [39] B. Wu, L. Zong, X. Yan, and C. G. Soares, "Incorporating evidential reasoning and TOPSIS into group decision-making under uncertainty for handling ship without command," *Ocean Eng.*, vol. 164, pp. 590–603, Sep. 2018.
- [40] F. Wang, H. Zhang, K. S. Li, Z. Y. Lin, J. Yang, and X. L. Shen, "A hybrid particle swarm optimization algorithm using adaptive learning strategy," *Inf. Sci.*, vols. 436–437, pp. 162–177, Apr. 2018.
- [41] H. Wang, W. Wang, H. Sun, and S. Rahnamayan, "Firefly algorithm with random attraction," *Int. J. Bio-Inspired Comput.*, vol. 8, no. 1, pp. 33–41, Feb. 2016.



XIUQIN PAN received the B.E. degree in electric technology and the M.E. degree in power system and automation specialty from Zhengzhou University, Zhengzhou, China, in 1994 and 1999, respectively, and the Ph.D. degree in control theory and control engineering from the Beijing Institute of Technology, in 2002. She is currently a Professor with the School of Information Engineering, Minzu University of China. Her current research interests include parallel algorithm and intelligent systems.



WANGSHENG ZHOU received the B.E. degree in automation from the Henan University of Science and Technology, Luoyang, China, in 2018. He is currently pursuing the master's degree with the School Software Engineering, Minzu University of China. His current research interests include parallel algorithm and intelligent systems.



YONG LU received the B.E. degree in mechanics from Zhengzhou University, Zhengzhou, China, in 1991, and the M.E. degree in national defense economy from the Military Economics Academy of PLA, Wuhan, China, in 2005. He is currently an Engineer with the School of Information Engineering, Minzu University of China. His research interests include parallel algorithm and intelligent systems.



NA SUN received the B.E. degree in electronic information engineering from the Hebei University of Technology, Tianjin, China, in 2004, the M.E. degree in communication and information system from Guizhou University, Guiyang, China, in 2007, and the Ph.D. degree in signal and information processing from the Beijing University of Posts and Telecommunications, in 2010. She is currently a Lecturer with the School of Information Engineering, Minzu University of China. Her current research interests include parallel algorithm and intelligent systems.

• • •