Ant colony algorithm based on data classification

Wanrong Tan¹, *, Jian Gao² and Jun Rao³

¹, ², ³ School of Mechatronics Engineering and Automation, Shanghai University, Shanghai, China

*Corresponding author e-mail: 1239454068@qq.com

Abstract. Aiming at the drawbacks of ant colony algorithm applied in the medium and large scale traveling salesman problem (TSP), this paper proposes an ant colony algorithm based on data classification (DACO), in which data classification method takes attraction degree as an important consideration. First, the initial data of the selected data set is divided into two categories - candidate class and elimination class according to the law of attraction degree. Then the elimination class is eliminated to reduce the search time. At the same time, the attraction degree can effectively guide the algorithm to search for the optimal solution when the previous pheromone heuristic effect is not significant. In addition, 3-opt local search algorithm is added to further optimize the solution. The experimental results show that the DACO has a better convergence and solution quality compared with other ant colony algorithms.

1. Introduction

Ant colony algorithm was first proposed by Italian scholar Dorigo and named ant system (AS). Researchers successively proposed the ant colony system (ACS) and the max-min ant system (MMAS) on the basis of AS[1]. TSP belongs to NP hard problem. The emergence of ant colony algorithm provides a new idea for solving the TSP. Scholars also put forward many improved algorithms. Mingle XU proposed a heuristic communication dual population ant colony algorithm without extra processing on pheromone matrix of each ant colony, saving time[2]. Xiwu Wang proposed an ant colony algorithm combining positive feedback and negative feedback to update pheromones, which enhanced the global search capability of algorithm[3].

An ant colony algorithm based on data classification is proposed in this paper. The data classification method is proposed to classify the selected TSP data set into candidate class and elimination class, narrowing the selection range, which is of great significance for solving large scale TSPs. At the same time, the attraction degree considers the pheromone information and city location comprehensively from the global perspective, guiding the algorithm to search for the optimal solution and obviously improving the convergence speed of DACO. In addition, the solution is optimized with 3-opt local search algorithm to improve the diversity of the solution. The DACO is tested with several medium and large TSP data sets in this paper. The experimental results show that compared with other improved algorithms, DACO achieves a balance between convergence and the diversity of solutions.
2. Basic composition of ACS

The most typical TSP in the NP hard problem was used to verify the improvement of the ACS in this thesis.

2.1. Path construction

In the ACS algorithm, if ants $m$ randomly select the starting point city from cities $n$ at the initial moment, then the ant $k$ in the city $i$ uses the pseudo-random ratio selection rule of equation (1) to select the city $j$ to be visited next:

$$j = \begin{cases} \arg \max_{s \in \text{allowed}} \{ \tau_{is}(t) \cdot \eta_{is}^{\beta} \} & , q \leq q_0 \\ \eta_{is} & , q > q_0 \end{cases}$$

(1)

Where $q_0$ is a constant between 0 to 1 and $q$ is a random variable subjected to a uniform distribution between 0 to 1. $\tau_{is}(t)$ is the pheromone concentration between city $i$ and $s$ in the t-th iteration, and $\eta_{is}$ is the reciprocal of the distance between city $i$ and $s$. allowed refers to the set of cities not visited by ant $k$ in this iteration. $J$ is equal to equation(2).

$$P_{ij} = \begin{cases} \frac{[\tau_{ij}(t)]^\alpha [\eta_{ij}]^\beta}{\sum_{s \in \text{allowed}} [\tau_{is}(t)]^\alpha [\eta_{is}]^\beta} & , j \in \text{allowed} \\ 0, & \text{otherwise} \end{cases}$$

(2)

Where $P_{ij}$ is the probability of ant $k$ from city $i$ to city $j$, $\alpha$ and $\beta$ are the relative importance of pheromone and heuristic information.

2.2. Phermone update

The pheromone update of ACS algorithm is divided into two parts: local pheromone update and global pheromone update. Every time an ant traverses all cities, the algorithm updates the pheromone locally on all the paths traveled by the ant in this iteration according to equation (3).

$$\tau_{ij}(t+1) = (1 - \zeta)\tau_{ij}(t) + \zeta \tau_0$$

(3)

Where $\tau_{ij}(t)$ is the pheromone concentration between city $i$ and city $j$ at t-th iteration. $\tau_0$ is the initial setting of the pheromone. $\zeta$ is the evaporation rate of local pheromone.

After all the ants have completed the city traversal in a single iteration, the global pheromone update is performed according to equation (4). This update process only applies to the edge of the global optimal solution.

$$\tau_{ij}(t+1) = (1 - \rho)\tau_{ij}(t) + \rho \Delta \tau_{ij}$$

(4)

$$\Delta \tau_{ij} = \begin{cases} 1/L_{gb} & , \text{if}(i,j) \in \text{BestTour} \\ 0 & , \text{otherwise} \end{cases}$$

(5)

Where $\rho$ is the evaporation rate of global pheromone. $\Delta \tau_{ij}$ is the pheromone increment, as shown in equation (5). $L_{gb}$ is the optimal path length found so far.

The initial pheromone between cities in ACS is evenly distributed, while the global pheromone evaporation rate is less than 1, making the pheromone update a slow accumulation process. Therefore, the "artificial ant" lacks strong global information guidance in selecting the next city in the early stage,
and it is difficult to find a better solution in the corresponding time, which directly affects the convergence of the algorithm.

3. 3-opt algorithm

In the improvement of ant colony algorithm, adding local search algorithm is a very important step. Among them, the most representative local search algorithm is 3-opt algorithm. The basic idea of the algorithm is to select the closed path \( L \) to be optimized first and then randomly delete the three edges. The path \( L \) is divided into three parts. And there are eight methods of connecting the three parts to form a new closed path \( L_0 \). The shortest of all possible paths is chosen as the optimized path \( L_1 \).

4. Ant colony algorithm based on data classification

4.1. Data classification method base on attraction degree

In order to solve the problem that ants cannot find a better solution due to the lack of corresponding global information guidance in the medium and large scale cities set in the early stage, this paper proposes a data classification method based on attraction degree. Attraction degree is a vector, as shown in definition 1 below:

**Definition 1:** Suppose that the set \( A = \{j | j = 1,2,\ldots,n\} \) represent unvisited cities \( n \) in this iteration.

If the current city of the ant is city \( i \), the attraction degree \( \vec{e}_{ij} \) of other cities \( j \) to the city \( i \) is shown in equation (6).

\[
\vec{e}_{ij} = \tau_{ij} (\eta_{ij})^\gamma e^{k\theta_{ij}}, \quad \gamma > 1
\]  

(6)

Where \( \eta_{ij} \) is the reciprocal of the distance between city \( i \) and \( j \), \( \tau_{ij} \) is the pheromone concentration between city \( i \) and city \( j \) in this iteration, \( \theta_{ij} \) is the angle between the straight line passing through city \( i \) and \( j \) and the positive direction of the horizontal axis of the rectangular coordinate system. \( k \) is an imaginary unit.

**4.1.1 Law of attraction degree.** Ants in the current city will be attracted by other unvisited cities to different degrees, and its value is given by equation (7). The attraction degree matrix \( W_i^T \) is established to represent the attraction degree of other cities \( j \) received by ant \( m \) in the current city \( i \) in the \( T \) iteration. The elements of each row in \( W_i^T \) separately represents the projection of attraction degree of other cities \( j \) on the rectangular coordinates axis with city \( i \) as the origin.

\[
W_i^T = \begin{bmatrix}
\tau_{i1}(\eta_{i1})^\gamma \cos \theta_{i1} & \tau_{i1}(\eta_{i1})^\gamma \sin \theta_{i1} \\
\tau_{i2}(\eta_{i2})^\gamma \cos \theta_{i2} & \tau_{i2}(\eta_{i2})^\gamma \sin \theta_{i2} \\
\vdots & \vdots \\
\tau_{ij}(\eta_{ij})^\gamma \cos \theta_{ij} & \tau_{ij}(\eta_{ij})^\gamma \sin \theta_{ij}
\end{bmatrix}
\]  

(7)

According to the orthogonal decomposition and synthesis method of vectors, the coordinates of total attraction degree are obtained by summing the elements of matrix \( W_i^T \) in columns, and then the candidate city class is determined by comparing the absolute value of the angle \( |\omega_h| \) with given angle threshold \( \omega_0 \). Candidate cities can be represented by set \( S \) according to equation (8):

\[
S = \{h | |\omega_h| \leq \omega_0, \ h \in N_j\}
\]  

(8)

Where \( h \) represents the selected candidate city, \( \omega_h \) is angle the between each attraction degree and
the total attraction degree, and \( N_j \) represents cities set not visited by ants.

The main idea of the law of attraction degree is to first establish the attraction degree matrix of the current city according to the definition of the attraction degree, and then sum the attraction degree from other unvisited cities at the current position through vector synthesis so as to obtain the total attraction degree. Finally, the candidate class data were determined by comparing \( |\omega_j| \) with \( \omega_0 \), and the data classification was completed, so as to shorten the time for the algorithm to select the city.

4.2. Algorithm framework of DACO

Table 1. Algorithm framework.

| 1. Initialization parameters of DACO |
|-------------------------------------|
| 2. While the termination condition is satisfied do |
| (1) Calculate the attraction degree of ant at the current position by Eq. (6) |
| (2) Establish the attraction degree matrix according to Eq. (7) |
| (3) Determine the candidate city set by the law of attraction degree |
| (4) Select the city to be visited next by Eq. (1) or (2) in candidate city set and construct the traversal path |
| (5) Local pheromone update |
| (6) Optimize the solution of this iteration with 3-opt algorithm |
| (7) Global pheromone update |
| (8) NC=NC+1 |
| 3. end while |

5. Experiment and simulation

This experiment was simulated in the IntelliJ IDEA2018.3.5 environment. In order to verify the performance of the DACO, we selected various medium and large scale TSPs, such as Att48, Pr76, KroD100, ch130, KroA150, etc. from the standard test library (TSBLIB). Each experiment was repeated 30 times, and each experiment completed 1500 iterations.

5.1. Parameters setting

In order to verify the influence of different angle thresholds in the algorithm, Eil51, Pr76, Pr107 were selected as representatives to conduct experiments. Experimental data are shown in table 2.

Table 2. The optimal path under different angle threshold in DACO algorithm

| Instance   | \( \omega_0 = \frac{5\pi}{9} \) | \( \omega_0 = \frac{\pi}{2} \) | \( \omega_0 = \frac{4\pi}{9} \) | \( \omega_0 = \frac{7\pi}{18} \) | \( \omega_0 = \frac{\pi}{3} \) |
|------------|-------------------------------|-------------------------------|-------------------------------|-------------------------------|-------------------------------|
| Eil51      | 428                           | 428                           | 429                           | 431                           | 431                           |
| Pr76       | 109086                        | 108159                        | 108159                        | 108804                        | 109126                        |
| Pr107      | 44301                         | 44301                         | 44385                         | 44385                         | 44436                         |
| KroE100    | 22141                         | 22068                         | 22068                         | 22068                         | 22810                         |
| KroA100    | 21285                         | 21285                         | 21285                         | 21577                         | 21373                         |
| KroB100    | 22197                         | 22139                         | 22234                         | 22197                         | 22234                         |

According to the table 2, if the angle threshold is too large, it fails to make full use of the information provided by the attraction degree. If it is too small, the randomness of ant selection will be reduced. When \( \omega_0 \) equal to \( \frac{\pi}{2} \), the algorithm achieves a balance between the use of global information and the random selection of ants. \( \omega_0 \) in this paper is set to \( \frac{\pi}{2} \). The remaining experimental parameters are set as shown in table 3.
Table 3. Parameters setting of DACO algorithm

| Instance | Num of ant | $\alpha$ | $\beta$ | $\zeta$ | $\rho$ | $\gamma$ |
|----------|------------|---------|---------|--------|--------|---------|
| Eil51    | 34         | 2       | 3       | 0.02   | 0.1    | 4       |
| Berlin52 | 35         | 2       | 3       | 0.02   | 0.1    | 4       |
| Att48    | 32         | 2       | 3       | 0.02   | 0.1    | 4       |
| St50     | 47         | 2       | 3       | 0.02   | 0.1    | 4       |
| Pr76     | 51         | 2       | 3       | 0.02   | 0.1    | 4       |
| Pr107    | 71         | 2       | 3       | 0.02   | 0.1    | 4       |
| Lin105   | 70         | 2       | 3       | 0.02   | 0.1    | 4       |
| Pr226    | 151        | 2       | 3       | 0.02   | 0.1    | 4       |
| Pr299    | 199        | 2       | 3       | 0.02   | 0.1    | 4       |
| ch130    | 87         | 2       | 8       | 0.02   | 0.1    | 4       |
| KroA100  | 66         | 2       | 8       | 0.02   | 0.1    | 4       |
| KroB100  | 66         | 2       | 8       | 0.02   | 0.1    | 4       |
| KroD100  | 66         | 2       | 8       | 0.02   | 0.1    | 4       |
| KroA150  | 100        | 2       | 10      | 0.02   | 0.1    | 4       |
| KroB150  | 100        | 2       | 10      | 0.02   | 0.1    | 4       |
| KroA200  | 133        | 2       | 12      | 0.02   | 0.1    | 4       |

5.2. Experimental data

The performance analysis of this algorithm mainly focuses on the best known solution (BKS), best solution found (Best), worst solution found (Worst), mean solution (Mean), error rate, standard deviation (SD) and convergence iteration (Convergence). The experimental results are shown in table 4, where the Error rate is calculated according to equation (9).

$$Error\ rate = \left( \frac{L_{\text{avg}} - L_{\text{min}}}{L_{\text{min}}} \right) \times 100\%$$  \hspace{1cm} (9)

$L_{\text{avg}}$ is the average of all the optimal solutions obtained in 30 experiments, and $L_{\text{min}}$ is the optimal solution for each instance given in TSPLIB.

Table 4. Experimental data of DACO algorithm

| Instance | BKS       | Best      | Worst     | Mean      | Error rate | SD         | Convergence |
|----------|-----------|-----------|-----------|-----------|------------|------------|-------------|
| Att48    | 33523     | 33523     | 33523     | 33523     | 0          | 0          | 45          |
| Eil51    | 426       | 426       | 429       | 428.1     | 0.49       | 0.3051     | 63          |
| Berlin52 | 7542      | 7544      | 7544      | 7544      | 0.02       | 0          | 24          |
| St50     | 675       | 677       | 685       | 678.3     | 0.48       | 3.2659     | 32          |
| Pr76     | 108159    | 108159    | 108275    | 108166.6  | 0.007      | 23.0688    | 75          |
| KroA100  | 21282     | 21285     | 21285     | 21285     | 0          | 0          | 54          |
| KroB100  | 22141     | 22139     | 22234     | 22187.0   | 0.20       | 41.9410    | 88          |
| KroD100  | 21294     | 21294     | 21375     | 21307.5   | 0.06       | 23.0295    | 205         |
| Lin105   | 14379     | 14382     | 14382     | 14382     | 0          | 0          | 31          |
| Pr107    | 44303     | 44301     | 44385     | 44307.0   | 0.009      | 22.4499    | 195         |
| Ch130    | 6110      | 6110      | 6179      | 6132.1    | 0.36       | 23.9052    | 179         |
| KroA150  | 26524     | 26524     | 26715     | 26547.2   | 0.08       | 59.0552    | 245         |
| KroB150  | 26130     | 26127     | 26436     | 26149.2   | 0.07       | 67.9687    | 221         |
| Pr226    | 80369     | 80370     | 80370     | 80370     | 0          | 0          | 106         |
| Pr299    | 48191     | 48194     | 48469     | 48261.9   | 0.15       | 87.4114    | 555         |
As can be seen from table 4, for instances Att48, Pr76, KroD100, Ch130 and KroA150 of TSP, the DACO obtained the optimal solution consistent with the official data. For large data sets Pr226 and Pr299, this algorithm obtained results very close to the official data. For large and medium scale instances such as KroB100, Pr107 and KroB150, the optimal solutions are 22139, 44301 and 26127 respectively, and even obtains better solutions than official data in TSPLIB.

![Figure 1](image)

**Figure 1.** Best tours for each instance found by DACO algorithm

5.3. Comparison between algorithms

In order to verify the improved effect of DACO algorithm on ACS algorithm and reduce the errors introduced by different programming languages, the ACS algorithm was re-implemented in IntelliJ IDEA2018.3.5. At the same time, in order to ensure the better performance of ACS algorithm, parameters setting are referred to in the literature [4].

### Table 5. Parameters setting of ACS algorithm

|   |   |   |   |   |
|---|---|---|---|---|
| $\alpha$ | $\beta$ | $\zeta$ | $\rho$ |
| 1  | 4  | 0.1 | 0.5 |

### Table 6. Comparison between DACO algorithm and ACS algorithm

| Instance | Best | Mean | Error rate | Convergence |
|----------|------|------|------------|-------------|
| Daco     | ACS  | Daco | ACS        | Daco        | ACS         |
| Eil51    | 428  | 428  | 428.1      | 431.2       | 0.49        | 1.22        | 63          | 544         |
| Berlin52 | 7544 | 7544 | 7544       | 7650        | 0.02        | 1.43        | 24          | 342         |
| Pr76     | 108159 | 108159 | 108166.6 | 109700.4 | 0.007 | 1.42 | 75 | 664 |
| KroA100  | 21285 | 21294 | 21285     | 21470.8    | 0     | 0.87 | 54 | 140 |
| KroB100  | 22139 | 22141 | 22187.0   | 22379.0    | 0.20  | 1.07 | 88 | 1165 |
| Pr107    | 44301 | 44346 | 44307.0   | 44389.0    | 0.009 | 0.19 | 195 | 216 |
| KroA150  | 26524 | 26774 | 26547.2   | 27118.8    | 0.08  | 2.24 | 245 | 1097 |
| KroB150  | 26127 | 26311 | 26149.2   | 26557.3    | 0.07  | 1.64 | 221 | 814 |
| Pr299    | 48194 | 48829 | 48261.9   | 49667.2    | 0.15  | 3.06 | 555 | 857 |

As can be seen from table 6, compared with the ACS algorithm, DACO algorithm greatly improves the solution quality and convergence speed. The variation of the optimal solution with the iterations is shown in Fig. 2. For Pr76 and KroA100, DACO algorithm finds the optimal solution in 75 and 54 iterations, while ACS algorithm needs 664 and 140 iterations to reach convergence. For KroA150 and KroB150, DACO algorithm can also find the optimal solution quickly, which proves that the
The introduction of data classification method is conducive to improving the performance of the algorithm.

![Figure 2](image)

**Figure 2.** Convergence curve of DACO and ACS for Pr76 and KroA100

| Method                        | Instance | Pr76 | KroA100 | Lin105 | KroB150 |
|-------------------------------|----------|------|---------|--------|---------|
| Proposed-method               | Avg      | -    | 21285   | 14382  | 26149   |
|                               | SD       | -    | 0       | 0      | 0       |
| Error(%)                      |          | -    | 0.49    | 0.49   | 0.07    |
| Modified RABNET-TSP[5]        | Avg      | 437.47| 7932.50 | -      | 21522.73| 14400.7 |
|                               | SD       | 4.21 | 277.25  | -      | 93.34   | 44.03   |
| Error(%)                      |          | 2.69 | 5.18    | -      | 1.13    | 0.15    |
| SA ACO PSO[6]                 | Avg      | 427.27| 7542    | -      | 21370.30| 14406.37| 26448.33|
|                               | SD       | 0.45 | 0.00    | -      | 123.36  | 37.28   | 266.76  |
| Error(%)                      |          | 0.30 | 0       | 0      | 0.41    | 0.19    | 1.22    |
| HACO[7]                       | Avg      | 431.20| 7560.54 | -      | -       | -       |
|                               | SD       | 2.00 | 67.48   | -      | -       | -       |
| Error(%)                      |          | 1.22 | 0.23    | -      | -       | -       |
| ACO+2opt[8]                   | Avg      | 439.25| 7556.58 | -      | 23441.80| -       | -       |
|                               | SD       | -    | -       | -      | -       | -       |
| Error(%)                      |          | 3.11 | 0.19    | -      | 10.15   | -       | -       |
| IVRS+2opt[8]                  | Avg      | 431.10| 7547.23 | -      | 21499.61| -       | -       |
|                               | SD       | -    | -       | -      | -       | -       |
| Error(%)                      |          | 1.20 | 0.07    | -      | 1.02    | -       | -       |
| ACO+ABC[9]                    | Avg      | 443.39| 7544.37| 700.58 | 22435.31| -       | -       |
|                               | SD       | 5.25 | 0.00    | 7.51   | 231.34  | -       | -       |
| Error(%)                      |          | 4.08 | 0.03    | 3.79   | 5.42    | -       | -       |
| ACO+Taguchi[10]               | Avg      | 435.19| 7635.02 | -      | 21567.34| 14475.05| -       |
|                               | SD       | 3.15 | 234.52  | -      | 102.26  | 98.38   | -       |
| Error(%)                      |          | 2.15 | 1.20    | -      | 1.34    | 0.67    | -       |
| PACO-3opt[11]                 | Avg      | 426.35| 7542    | 677.85 | 21326.80| 14393.00| -       |
|                               | SD       | 0.49 | 0.00    | 0.99   | 33.72   | 19.76   | -       |
| Error(%)                      |          | 0.08 | 0.00    | 0.42   | 0.21    | 0.10    | -       |
| PCCACO[4]                     | Avg      | 426.73| 7542    | -      | 21419.33| -       | 26241   |
|                               | SD       | -    | -       | -      | -       | -       |
| Error(%)                      |          | 0.23 | 0.00    | 0.47   | -       | 0.42    |

As can be seen from table 7, with the increase of the number of cities, the performance of other improved algorithms decreases, while the performance of the DACO is more excellent. For such medium-sized data sets as KroA100, Lin105 and KroB150, DACO algorithm can obtain the optimal solution with very small error rate and the average solutions are 21285.0, 14382.0 and 26149.2.
6. Conclusion and future work

An ant colony algorithm based on data classification is proposed and used to solve the TSP in this paper. The data classification method plays an important role in the effectiveness of each step of ant selection. The attraction degree is taken as the consideration basis of global information to reduce the probability of the algorithm falling into local optimum. In addition, the solution is optimized locally by 3-opt method. The experimental results show that the proposed ant colony algorithm based on data classification is excellent in solving TSP and satisfactory results have been obtained.

For large TSP problems, how to reduce the time complexity of solution is still a problem. In the following work, we will study the influence of the parameters setting of the algorithm on the experimental results and solve more complex problems.

References
[1] Li wen. Ant colony optimization algorithm and its application research [D]. Hunan university, 2005.
[2] M.-L. Xu, X.-M. You, and S. Liu, "A novel heuristic communication heterogeneous dual population ant colony optimization algorithm," IEEE Access, vol. 5, pp. 18506-18515, 2017.
[3] X. Wang, Y. Wang, Y. Wang and Y. Jin, "Notice of Retraction: A new ant colony optimization algorithm for TSP," 2013 International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering (QR2MSE), Chengdu, 2013, pp. 2055-2057.
[4] H. Zhu, X. You and S. Liu, "Multiple Ant Colony Optimization Based on Pearson Correlation Coefficient," in IEEE Access, vol. 7, pp. 61628-61638, 2019.
[5] T.A.S. Masutti, L.N. de Castro, A self-organizing neural network using ideas from the immune system to solve the traveling salesman problem, Inf. Sci. 179, 2009, pp. 1454-1468.
[6] S.-M. Chen and C.-Y. Chien, “Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques,” Expert Syst. Appl., vol. 38, no. 12, pp. 14439-14450, 2011.
[7] W. Junqiang and O. Aijia, “A hybrid algorithm of ACO and delete-cross method for TSP,” in Proc. Int. Conf. Ind. Control Electron. Eng., Jun. 2012, pp. 1694-1696.
[8] K. Jun-Man and Z. Yi, “Application of an improved ant colony optimization on generalized traveling salesman problem,” Energy Procedia, vol. 17, pp. 319-325, Apr. 2012.
[9] M. Gunduz and M. S. Kiran, and E. Özceylan, “A hierarchic approach based on swarm intelligence to solve the traveling salesman problem,” Mathematics, vol. 23, pp. 215-235, Mar. 2015.
[10] M. Peker, B. Şen, and P. Y. Kumru, “An efficient solving of the traveling salesman problem: the ant colony system having parameters optimized by the Taguchi method,” Turkish J. Electr. Eng. Comput. Sci., vol. 21, pp. 2015-2036, May 2013.
[11] G. Şaban, M. Mostafa, B. O. Kaan, and H. Kodaz, “A parallel cooperative hybrid method based on ant colony optimization and 3-Opt algorithm for solving traveling salesman problem,” Soft Comput., vol. 22, no. 5, pp. 1669-1685, Mar. 2018.