第一页为封面页

参赛队员姓名： 曹熙媛

中学： 华南师范大学附属中学

省份： 广东省

国家/地区： 中国

指导教师姓名： 宋亮， 杨晓安

指导教师单位： 中山大学数学学院, 广东华南师大附中

论文题目： Stochastic Bionic Model of Physarum Polycephalum Multi-Source Food Foraging for Minimum Spanning Trees and Graph Clustering

Stochastic Bionic Model of *Physarum Polycephalum* Multi-Source Food Foraging for Minimum Spanning Trees and Graph Clustering

Xiyuan Cao

Abstract:

This paper investigates a bionic computing algorithm based on swarm intelligence that simulates the intelligent behavior of the multi-source food foraging of *Physarum polycephalum*. An algorithm called PMA-MST[1] is proposed that can determine the minimum spanning tree of a network. After statistical principles are introduced to PMA-MST, it can perform clustering based on the determined minimum spanning tree. The proposed PMA-MST algorithm is embedded with randomness, and its uncertainty property leads to the parallelism of the algorithm, which is beneficial for improving computational efficiency and determining the globally optimal solution. Several examples, including those on power grid optimization and test datasets, demonstrate that the method is effective, stable, and robust.

Key words: *Physarum polycephalum*, PMA-MST algorithm, randomness, minimum spanning tree, graph clustering, intelligent computing

目录

[1] Physarum Modeling Accelerated- Minimum Spanning Tree algorithm

1. Introduction

Abundant phenomena in the natural world can be imitated by humans. For example, people study bionic algorithms that imitate the behaviors of groups of creatures in nature. Many algorithms have been developed, including ant colony optimization, particle swarm optimization, the wolf pack algorithm, artificial bee colony algorithm, glowworm swarm optimization, the artificial fish swarm algorithm, and bacteria foraging optimization, to simulate swarm intelligence (Beni G, Wang J. 1989)[2]. This kind of bionic algorithm is sometimes called nature-inspired computing (NIC), and it has a wide range of applications (Shadbolt, 2004; Lilakari, Grzegorz Rozenberg, 2008). NIC can solve many problems such as complex optimization, neural networks, the traveling salesman problem, the knapsack problem, clustering, and classification, and is used in many fields such as the life sciences, road network planning, power system dispatching, pattern recognition, and communication networks. In the process of exploring the phenomena of life, researchers are constantly constructing new models to imitate its exquisiteness, and the *Physarum polycephalum* bionic algorithm is one of them.

P. polycephalum is a single-celled organism with no advanced nervous system or even neurons. However, a large number of *P. polycephalum* can self-organize to form a foraging network without any global information coordination. *P. polycephalum*'s behavior mechanism can be imitated to solve the maze problem (Toshiyuki Nakagaki, 2000). Tero (2010) even reported that *Physarum* can form a foraging network similar to the actual railway network topology in Tokyo. In addition, *P. polycephalum* can also solve geometric problems and logic gate problems, identify directions, and exhibit spatial and behavioral memory[3]. In addition, it can effectively simulate the American highway network (Adamatzky A and Jones J, 2010), and solve complex problems in wireless sensor networks (Tsompanas M-A I, Mayne R, Sirakoulis G C and Adamatzky A I 2015, 2015). The extraordinary and seemingly intelligent *P. polycephalum* has been studied since 1822 (von Schweinitz L D), and is still a popular topic for cell biologists, chemists, physicists, computer scientists, and many other scientists in different fields.

The minimum spanning tree (MST), also known as the minimum cost spanning tree, refers to the spanning tree with the lowest cost generated in an undirected connected network[4] (that is, a tree with the shortest total path or the smallest weight). The MST is of great significance in a network graph, and it represents the most effective structure contained in that network.

Clustering is the process of dividing a set of objects into subsets to determine the categories to which these objects belong. The classification for clustering is unknown, and the clustering process is performed using unsupervised learning. Clustering is an important research topic in pattern recognition and data analysis, and has been applied in many fields. Graph clustering is clustering

[2] Swarm intelligence research is a branch of artificial intelligence, which mainly studies how simple individuals achieve orderly and structured organization through simple interactive behaviors and then produce meaningful behaviors and meaningful results. For example, the principle of an artificial neural network is that a large number of simple neurons cooperatively work through connections and information exchange. As a result, a structuralist artificial intelligence method with a brain-like function is produced.

[3] http://www.im.cas.cn/kxcb/wswdjt/201309/t20130926_3939681.html

[4] A subgraph generated from a network graph composed of N nodes contains all N vertices and n − 1 edges in the original network graph. Each edge of spanning tree T is given a weight (the length of each edge), and the sum of the weights is called the weight of the tree. The spanning tree with the smallest total weight value is called the MST of the graph.

based on graphs. Here, a graph is a network graph composed of vertices and edges, and graph clustering divides a network graph into several subgraphs. Many application scenarios in modern society include networks.

Because of *P. polycephalum*'s outstanding intelligence, this study employs the concept of stochasticity and proposes a bionic intelligent algorithm that imitates the multi-source food foraging of *P. polycephalum*. The main work includes the following contributions: a PMA algorithm to accelerate the PM algorithm, which is the classic model for *P. polycephalum*; the PMA-MST algorithm, which is based on a stochastic method to construct network MST; and a clustering method based on PMA-MST. The algorithm presented in this paper can solve both the MST and graph clustering problems effectively, as demonstrated by the numerical experiments.

2. Related Work and the PM Algorithm

2.1. Behavioral Characteristics of *P. Polycephalum*

P. polycephalum, which is a multinucleated single-celled organism also known as slime mold, is a type of amoeba. Its life cycle goes through many forms, such as protoplasm, mycelia, zoospores, sporangia, and zygotes (Fig.1). It becomes a sclerotium when the environment is not suitable for survival, and it turns into a protoplasm when the environment improves. *P. polycephalum* has good deformability, and it can form its own foraging network without any global information coordination using amorphous finger-like, leaf-like, or needle-like protrusions on the body surface. According to biologists, this is because its cells exchange different chemical signals according to the conditions of light, pressure, temperature, humidity, and substance exchange in the environment. This then produces chemical reactions, which lead to oscillations and other phenomena. In *P. polycephalum*'s foraging process, pipelines for transporting nutrients are formed, and the diameters of the pipelines increase and decrease with the change of nutrient sources (Christina Oettmeier, Klaudia Brix, Hans-Günther DBE Reiner, 2017). In addition, this experience is saved in memory (Mirna Kramar, Karen Alim, 2021). The rhythmic shrinking of cells causes the protoplasm to stream and then spread in different directions, evolving with changes in the environment. Some transportation pipelines become thicker, while others shrink, until the optimal foraging network is finally formed.

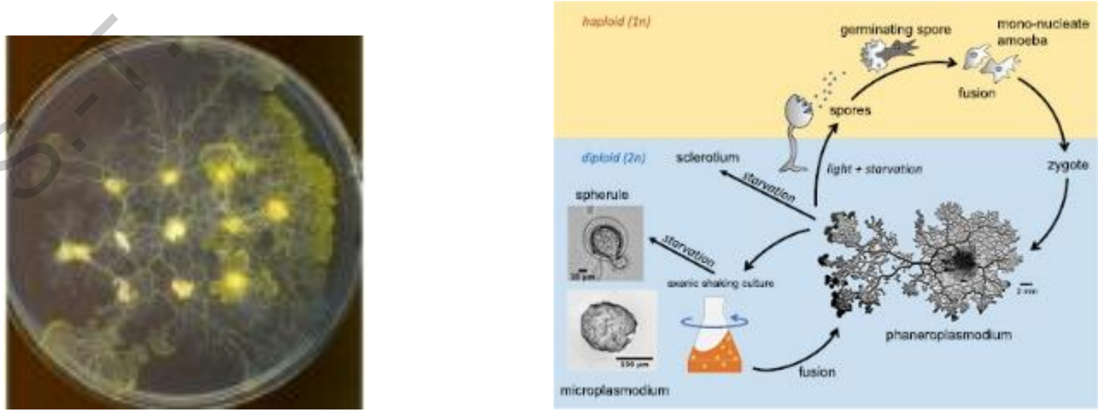


Fig. 1 Protoplasm morphology of *P. polycephalum* and its life cycle
(Christina2017; Andrew Schumann, Krzysztof Pancerz, 2016.)

T. Nakagaki et al. (2000) found that, on a maze formed by an agar surface, *P. polycephalum* can determine the shortest path between two the positions of nutrient agar blocks placed in the maze， as shown in Fig.2. That is, *P. polycephalum* can determine the shortest path that connects two food sources in a network. In 2010, a Japanese research team found that *Physarum* can connect food sources that represent 36 geographical locations in Tokyo. The resulting nutrient network was not only similar to the actual railway network structure, but also consistent in cost and transportation efficiency. Kramar and Alim (2021) studied the ability of *P. polycephalum* to find food through memory. It was found that *P. polycephalum* can store empirical information during its growth.

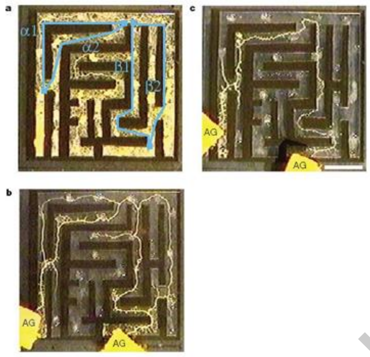


Fig. 2 Maze experiment. First, a maze pattern is constructed on the surface of the agar. Second, the *P. polycephalum* fragment is placed in the maze. *P. polycephalum* fragments begin to grow and spread outward, filling the maze. Nutritional food sources were then placed at the entrance and exit of the maze (AG). Here, α_1 and α_2 ($\alpha_1 > \alpha_2$) as well as β_1 and β_2 ($\beta_1 \approx \beta_2$) are the alternative competitive paths in the maze. Over time, some pipelines in the network of *P. polycephalum* gradually shrank and disappeared, forming the final stable network. Many experiments verified that the pipes with shorter paths were retained, longer pipes disappeared, and the probabilities that pipelines with similar path lengths remained were equal.

P. polycephalum, which has been living on the earth for hundreds of millions of years, has no brain or nerve cells and is not a higher organism. Therefore, we cannot attribute its intelligent behavior to the nervous system, as we can in higher organisms. However, even without a central control mechanism, an efficient network can be constructed during the foraging process. The constructed network can change its shape according to the environment and achieve economic and efficient self-optimization, which demonstrates the intelligent behavior of *P. polycephalum*. Its evolutionary mechanism needs further exploration.

Researchers have abstracted the behavioral characteristics of *P. polycephalum* to design rules and develop algorithms that imitate the evolutionary behavior of *P. polycephalum*. These algorithms include the PM, Oregonator, CELL, and Jones models. The most influential one is the PM model proposed by Tero et al., which is described in detail in the next section. In addition, the Oregonator model (Adamatzky,2009) simulates the oscillation phenomenon in the Belousov–Zhabotinsky reaction process, which describes the growth and foraging behavior of *P. polycephalum* through changes in reactant concentration. Adamatzky suggested that *P. polycephalum* can be thought of as a biological computer. The CELL model is described as an aggregation of lattice sites in a particular state (Gunji,2008), and it can be migrated and modified to simulate the evolution process of *P. polycephalum*. The Jones model consists of a number of agent particles with perceptive ability. According to the concentration of food chemicals, they gradually move from the initial random distribution and gather at the food source, finally forming a food source network that reflects the emerging group behavior of agent particles.

In a word, because of the surprising behavior of *P. polycephalum*, increasingly more scientists have desired to imitate the growth and behavior of *P. polycephalum* from the group perspective (the PM and Oregonator models) or individual perspective (the CELL and Jones models). The corresponding studies solve some problems such as the shortest path in a network and the traveling salesman problem. The bionic computation of *P. polycephalum* is different from traditional artificial intelligence (such as neural networks) and is instead a natural biological swarm intelligence method.

2.2. PM Algorithm

In 2000, Nakajaki discovered that *P. polycephalum* could solve the maze problem with the shortest path in a few hours. Tero et al. (2006) presented PM algorithm[5], a mathematical model for solving maze and building network to simulate *P. polycephalum*'s behavior based on Hagen–Poiseuille's law and Kirchhoff laws (Tero, 2006,2007,2010). Most modern methods for studying the shortest distance of networks using bionic *P. polycephalum* models are based on this method.

In the maze experiment, *P. polycephalum* first grows and spreads over the whole network. The protoplasm of the cells oscillates according to the conditions in the environment, and then some nutrient-channeling tubes shrink. In the process of pipe convergence and expansion, there is a positive feedback relationship between the pipeline radius and its flux. That is, an increase in flux leads to an increase in pipe radius, which in turn continues to cause increases in the flux in the pipeline. On the contrary, if the radius decreases, the flux decreases. The radius of a pipeline is related to the conductivity D, and the flux is denoted by Q.

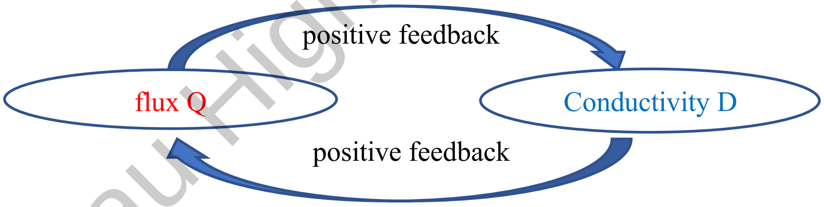


Fig. 3 Positive Feedback Principle

The PM algorithm, which is based on the principle of positive feedback, is as follows:

Conductivity D is related to the pipe radius as $D_{ij} = \frac{\pi r^4}{8\eta}$, where r is the pipe radius, η is the viscosity of the fluid, which is a constant. In the network, the pressure at point i is p_i and the pressure at point j is p_{j} (here, i and j represent the two nodes of an edge). Finally, L_{ij} denotes the length of edge ij. At the n-th time step, we have

$$Q_{ij}(n) = \frac{D_{ij}(n)}{L_{ij}}(p_i - p_j). \qquad (1)$$

This is Poiseuille's law, which describes the relationship between flux and conductivity. There is only one source node *1* and one sink node *2* in the network. The sum $\sum Q_j$ of each node j is equal to flux I_j at this node. Therefore, $\sum Q = I_0$ at source node *1* and $\sum Q = -I_0$ at sink node *2*. For other nodes in the network, we have $\sum Q = 0$. In addition, I_0 is constant (e.g., $I_0 = 1$). This gives us the

[5] *Physarum* solver - Modeling

following equation.

$$\sum_j \frac{D_{ij}(n)}{L_{ij}}\left(p_i - p_j\right) = \begin{cases} I_0 & for\ j = 1 \\ -I_0 & for\ j = 2 \\ 0 & otherwise \end{cases} \tag{2}$$

At the *n*+1-th step, the change is conductivity *D* is related to function $f\left(\left|Q_{ij}\right|\right)$, which is optional. The adaptation equation is expressed as follows.

$$\frac{dD_{ij}(n+1)}{dt} = f\left(\left|Q_{ij}\right|\right) - D_{ij}(n+1) \tag{3}$$

Then, the conductivity at the *n*+1-th step is expressed as

$$D_{ij}(n+1) = \left(f\left(\left|Q_{ij}(n)\right|\right) - D_{ij}(n+1)\right) \times \triangle\, t + D_{ij}(n), \tag{4}$$

where $f\left(\left|Q_{ij}\right|\right) = \left|Q_{ij}^{\mu}\right| = \left|Q_{ij}\right|$ (μ=1[6]).

Let Δt = 1. Then, we have the following.

$$D_{ij}(n+1) = \left(\left|Q_{ij}(n)\right| - D_{ij}(n+1)\right) \times \triangle\, t + D_{ij}(n)$$

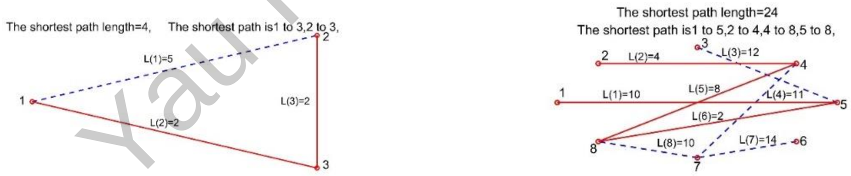
$$\Delta t{=}1 \qquad = \left(\left|Q_{ij}(n)\right| - D_{ij}(n+1)\right) + D_{ij}(n)$$

$$\therefore \quad D_{ij}\,(n+1) = \frac{1}{2}\left(f\left(\left|Q_{ij}(n)\right|\right) + D_{ij}(n)\right)$$

$$= \frac{1}{2}\left(f\left|Q_{ij}(n)\right| + D_{ij}(n)\right) \tag{5}$$

When performing this calculation, the initial D_{ij} is set to a random number; L_{ij} is assigned; $I_0 = 1$; and $p_2 = 0$.

In this paper, the two examples are given, and the results are shown in Fig. 4. In example 1, the shortest path from 1 to 2 is finally chosen as 1→3→2, which has a total length of 3, which is less than the complete path (of length 5) from 1 to 2.


The shortest path length=4, The shortest path is1 to 3,2 to 3,
1
2
3
L(1)=5
L(2)=2
L(3)=2
The shortest path length=24
The shortest path is1 to 5,2 to 4,4 to 8,5 to 8,
1
2
3
4
5
6
7
8
L(1)=10
L(2)=4
L(3)=12
L(4)=11
L(5)=8
L(6)=2
L(7)=14
L(8)=10

Example 1: Three nodes and three edges

b. Example 2: Eight nodes and eight edges

Fig. 4 Two examples of the PM algorithm, where node 1 is the source node, and node 2 is the sink node

PM is used to establish a multi-food source model according to how *P. polycephalum* forms a foraging network for multiple (three or more) food sources. Only one source node and one sink node can be selected each time. However, the selection is random. That is, the whole process is divided into multiple time steps, and the source point and sink point are randomly selected each time. If sink *j* is selected first, the probability that other points *i* of the whole network are selected as source points is inversely proportional to the distance between node *i* and sink *j*. That is, the farther the node is, the more likely it is to become a source node.

[6] When the value is greater than 1, the convergence is accelerated and the shortest path is obtained; when the value is less than 1, the convergence is also accelerated (Zhou Huan, 2012).

2.3. Graph Clustering Method

Clustering divides objects into different classes according to certain standards. Individuals in the same class have higher similarity, whereas individuals in different classes have relatively lower similarity[7]. There are many clustering algorithms, but each has its own advantages and disadvantages. Hence, there is no all-purpose way to solve all problems without any obstacles.

The most classical clustering algorithm is K-means (J. B. MacQueen,1967), which is a clustering algorithm based on distance. Its main idea is to determine *k* class centers of data so that the sum of squares of the distances between data points in dataset and the center of the class is minimized. Then, the dataset is divided into *k* classes. Each class is called a cluster[8]. The K-means method suggests that distance is often the basis of classification in clustering.

Network graph clustering refers to the division of vertices in a network graph. An edge of the network, as a weight (edge length), can indicate the similarity between two nodes. Network graph clustering divides the network graph into several subgraphs. The optimal partitioning problem of a graph is an NP-hard problem.

In 1971, Zahn proposed the MST-based clustering algorithm. MST-based clustering searches for an MST in an undirected graph G=[V, E] (here, V denotes vertices, and E denotes edges), of which the edge length (e.g., Euclidean distance) represents the relationship between nodes. Then, the algorithm determines a threshold and removes the edges with weights that are greater than that threshold in the MST. This generates a forest, and every tree in the forest is regarded as a cluster[9]. The clustering algorithm of MST can detect clusters with arbitrary shapes (Jothi, Mohanty, & Ojha, 2018)[10].

Prim's and Krustal's algorithms are commonly used to search for an MST, but these two algorithms start from local information and search gradually. The difference is that Krustal's algorithm sorts the edges, but Prim's algorithm does not. In this way, Krustal's algorithm makes use of global information.

3. Proposed PMA-MST Algorithm

In this paper, MST and clustering algorithms for multi-source foraging of *P. polycephalum* are given as a comprehensive combination of the PM algorithm and MST-based graph clustering method.

PM-MST algorithm, which is a *P. polycephalum* bionic algorithm, is described in this section. We first present an accelerated PMA algorithm. Then, the PMA multi-source foraging algorithm

[7] https://www.cnblogs.com/U940634/p/9758300.html

[8] The method is realized by the following steps: first, *k* data points are randomly selected as the initial cluster centers, the distance is calculated between each point in the data set and the selected cluster center, and each sample point is allocated to the corresponding cluster according to the principle of nearest distance. This is cyclically iterated using the average value of each data object contained in each cluster as the new cluster center, until the criterion function converges (the mean square error is usually used as the criterion, that is, the sum of squares of the distance from each point to the nearest cluster center is minimized).

[9] https://blog.csdn.net/sanshiguan/article/details/81038366

[10] Many clustering algorithms quantify the similarity between objects based on distance (such as Euclidean distance). However, when this method is used, only spherical clusters or convex clusters with similar size and density can be found. In many scenarios, by contrast, the shape of class clusters may be arbitrary. https://www.cnblogs.com/LittleHann/p/6595148.html

that models the *P. polycephalum* multi-source environment is described. The idea of randomness is incorporated into the PM-MST algorithm, which enables a parallel[11] search of the MST between any two nodes in the network. Finally, the MST-based clustering algorithm is described.

3.1. PMA: PM Accelerated Algorithm

（1）.　Principle of the Algorithm

The positive feedback principle of PM algorithm is that the flux in the pipeline is inversely proportional to the distance L between nodes and directly proportional to the conductance D, as shown in Eq. (1). In this paper, we let the denominator in Eq. (1) be L^{α}, where the acceleration parameter α > 1. After testing, in general, α is set to no more than 10. When α>10, the calculated results will not change much. Parameter α also has other functions that are discussed in Section 5.

Function L^{α} is a monotonically increasing function that makes the algorithm converge faster. PMA is calculated using the following equation.

$$Q_{ij}(n) = \frac{D_{ij}(n)}{L_{ij}{}^{\alpha}}\left(p_i - p_j\right) \tag{6}$$

（2）.　Algorithm Verification

Some examples were used to compare the performance of PM and PMA. The results show that PMA converges faster when searching the shortest path between two nodes. The aim in example 3 (Fig.5) is to determine the shortest distance from node 1 to node 2 in a network graph with four edges and four nodes. The shortest path is 1→3→2, and the total path length is 6. The results are the same when α = 1 and α = 5, but when α = 5, the algorithm converges faster than when α = 1. Under the same total number of iterations, which is 25, the maximum difference between the D and Q of each edge in the last two steps when α = 5 is smaller than when α = 1, which indicates that larger values of α lead to faster convergence. Therefore, the convergence of the PMA algorithm is faster.

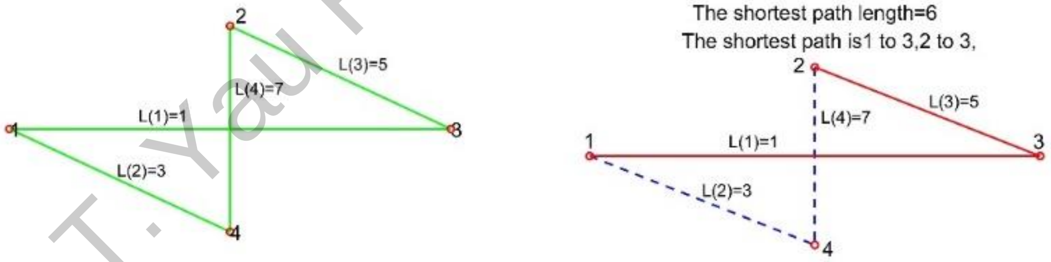
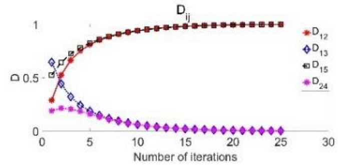
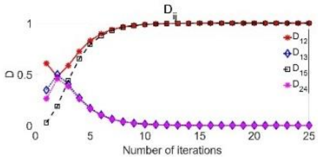
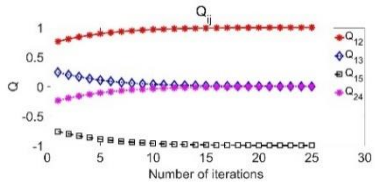
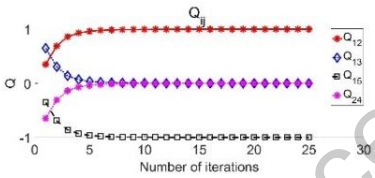

2
L(3)=5
L(4)=7
L(1)=1
1
3
L(2)=3
4
The shortest path length=6
The shortest path is1 to 3,2 to 3,
2
L(3)=5
L(4)=7
1
L(1)=1
3
L(2)=3
4

Fig. 5 Example 3

[11] Randomness leads to uncertainty, which is related to the inherent parallelism of the algorithm. Wang Peng et al. (2010) believes that a natural algorithm introduces probability, and the parallelism of a natural algorithm originates from the uncertain searching systems constructed by it, which realizes a parallel search of the solution space. However, the cost paid is that finding the optimal solution of the problem cannot be ensured.

Table 1 Larger values of α result in faster convergence

	α=1	α=5
D	 max (|D$_n$- D$_{n-1}$|)= 0.0027	 max (|D$_n$- D$_{n-1}$|)= 7.2396e-07
Q	 max(|Q$_n$- Q$_{n-1}$|)= 0.0016	 max (|Q$_n$- Q$_{n-1}$|)= 1.2484e-07

3.2. PMA-MST Intelligent Bionics Algorithm

（1）. Principle of the Algorithm

The MST is characterized by low cost, where the cost is the total length (weight) of the MST. Using the PM algorithm, if the shortest path between any two nodes is calculated first, for *n* nodes, it should be performed $C_n^2 = \frac{n(n-1)}{2}$ times. Then, the minimum cost tree is selected through extra evaluations and comparisons.

The aim of this study is to determine a MST with the minimum calculation cost.

The MST is characterized by low cost, and hence the algorithm proposed in this paper makes the following basic assumptions: 1) the minimum cost refers to the shortest path between any two nodes on average and 2) if the cost is the global minimum, there will be no redundant edges (because redundant edges increase cost) and hence there will be no loops.

In this study, we introduce the idea of stochasticity into PMA-MST. In every simulation, half of the nodes in the network are randomly set as the source nodes and the other half as the sink nodes. The source nodes and sink nodes are different every time. Intuitively, this is like holding half of an amount of seeds, and then sowing them randomly on the grid nodes, many times.

If different nodes are used as source nodes and sink nodes, the PMA-MST algorithm will automatically determine the most economical flow allocation structure. If the flux of an edge is 0, this edge is a costly edge. If there still is flow in an edge, that is, the flux is greater than 0, this edge is a low-cost edge, and it is a final selection after the comprehensive optimization[12]. Every node of the whole network is a food source, which is the multi-food source foraging problem. In each calculation, the source node and sink nodes in the network are paired by themselves. Repeating these operations is the same as finding the average shortest distance between any two nodes of the network. Because the algorithm automatically allocates flow to each edge every time, it can ultimately automatically decide which edge needs to be kept and which edge needs to be removed. Therefore, the PMA-MST algorithm is intelligent.

[12] Comprehensive optimization refers to a method that uses any two pairs of nodes as the source node and sink node, and the final edges selected by the algorithm are those edges that result in the smallest cost distance as a whole. Of course, we use the stochastic principle for simulation and calculation.

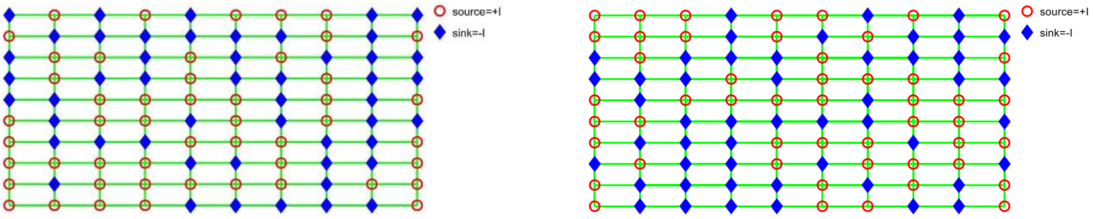


Fig. 6 Different sources and sinks randomly are generated by two different simulations. In the graph, the red circle indicates the source node and the blue diamond indicates the sink node. Each source node has a positive flow +I, and the sink node has a negative flow −I. The total sum of flow in the graph is 0. The results of two runs are shown, and the source nodes and sink nods are different each time.

If there is an odd number of nodes in the network, we set $\frac{(n-1)}{2}$ as source nodes, $\frac{(n-1)}{2}$ as sink nodes, and the remaining node is equal to 0.

The algorithm is run n_k times, then we sum all Q_k (k = 1 … n_k) of each edge for each trial, and calculate its average and round it. The remaining edges are the edges of the minimum path of the network, that is, the MST.

（2）. PMA-MST Process

The process of PMA-MST is as follows:

1. Set the number of simulation times n_k;
2. Assignment: half of the nodes in the network are set as source nodes with inflow I_0, and the other half are set as sink nodes with outflow $-I_0$;
3. Run the PMA algorithm and record the flux Q_{ijk} of each edge every time (*i, j* is the number of network nodes and *k* is the number of simulation steps);
4. At the end of simulation, the absolute value of flow $|Q_k|$of each edge is totaled[13] to obtain the total flux s_{ij} of each edge;
5. Q_{ij}^{mean}=s_{ij}/ n_k, and round the result.
6. For each edge, ifQ_{ij}^{mean}= 0, the edge is not preserved; otherwise, Q_{ij}^{mean} = 1 and the edge is preserved.

3.3. Graph Clustering Algorithm Based on PMA-MST

（1）. Principle of the Algorithm

In this algorithm, we cut the edges exceeding the threshold θ in the generated MST. Threshold θ here is adjustable and is defined as follows:

$$\theta = \bar{L} + \beta \cdot \sigma \quad . \tag{7}$$

Here, $\bar{L}$ is the average length of the MST, σ is the standard deviation of the length of the MST, and β is its coefficient.

In the theory of statistics, the mean value μ and standard deviation σ are very important. For instance, $\mu+\sigma$ contains 68% of the data, $\mu+2\sigma$ contains 95% of the data, and $\mu+3\sigma$ contains 99.7% of the data. Therefore, the standard of clustering can be adjusted. When β is smaller, more edges are

[13] Because flux Q can be negative, to prevent the positive and negative offset, the absolute value is taken.

removed, and classes are more divided. When β is larger, fewer edges are cut, and the classes are less divided.

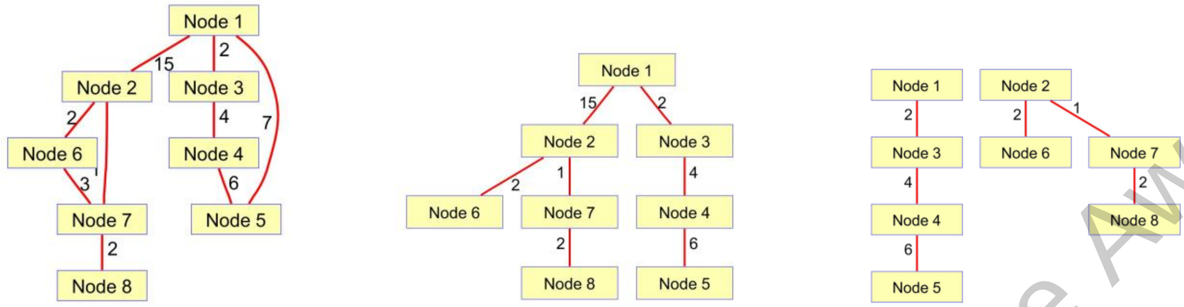
（2）. Graph Clustering Algorithm Process

The graph clustering algorithm process based on PMA-MST is shown below.

```
(Initialization)
  1. Random seeding: randomly select network nodes as source nodes and sink nodes;
     (For a network with $n$ nodes, when $n$ is an even number, $n/2$ nodes are set as the source nodes and $n/2$ nodes are set as the sink nodes; When $n$ is an odd number, $(n-1)/2$ nodes are set as the source nodes, $(n-1)/2$ nodes are set as the nodes, and the flow at the remaining node is 0. )
  2. Let $I_0 = 1$ at the source node and $-I_0 = -1$ at the sink node;
(Calculate the MST: PMA-MST)
  1. PMA algorithm
     （1）. Assign a random number to the conductivity $D_{ij}$ of each edge ($i, j$ is the node number); $p_2 = 0$;
     （2）. According to Eq. (2) at each node $j$, $\sum_j \frac{D_{ij}(n)}{L_{ij}{}^{\alpha}}(p_i - p_j) = \begin{cases} 1 \\ -1 \\ 0 \end{cases}$, solve for the pressure $p_j$ at each node $j$;
     （3）. According to Eq. (1), $Q_{ij}(t) = \frac{D_{ij}(t)}{L_{ij}{}^{\alpha}}(p_i - p_j)$, calculate the flow $Q_{ij}$ of each edge;
  2. Repeat PMA;
  3. Record the flux $Q_{ij}^t$ of each edge at the end of each optimization.
  4. Calculate $s_{ij}=\sum_{t=1}^{n_k}|Q_{ij}^t|$, where $n_k$ is the number of random seeds sown;
  5. Calculate $Q_{ij}^{mean}=s_{ij}/n_k$ and round it to the nearest integer to obtain the final result $Q_{ij}^{end}$;
  6. If $Q_{ij}^{end} = 0$, the $ij$-th edge is removed; if $Q_{ij}^{end} = 1$, then this edge is retained;
(Clustering)
  1. Calculate threshold $\theta$ using
         $\theta = \bar{L} + \beta\sigma$
  2. Remove the edges exceeding $\theta$ in the MST.
  3. Generate the final cluster.
```

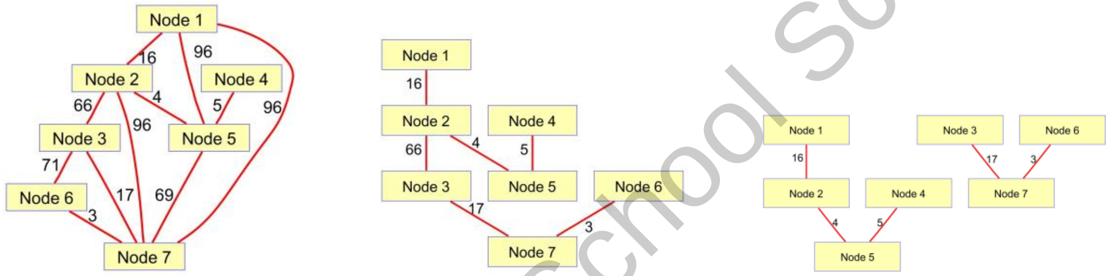
4. Numerical Experiment

This paper presents three examples to demonstrate the use of the PMA-MST graph clustering algorithm. In the example 4, the original network has eight nodes and nine edges, as shown in Fig. 7; in the example 5, the original network has seven nodes and 11 edges, as shown in Fig. 8; in the example 6, the original network has 25 nodes and 113 edges, as shown in Fig. 9. Each figure contains the initial network graph (Figs. 7a–9a), the MST obtained by PMA-MST (Figs. 7b–9b), and the clustering result (Figs. 7c–9c). Numerical experiments show that the algorithm is effective and can obtain the MST and clusters.



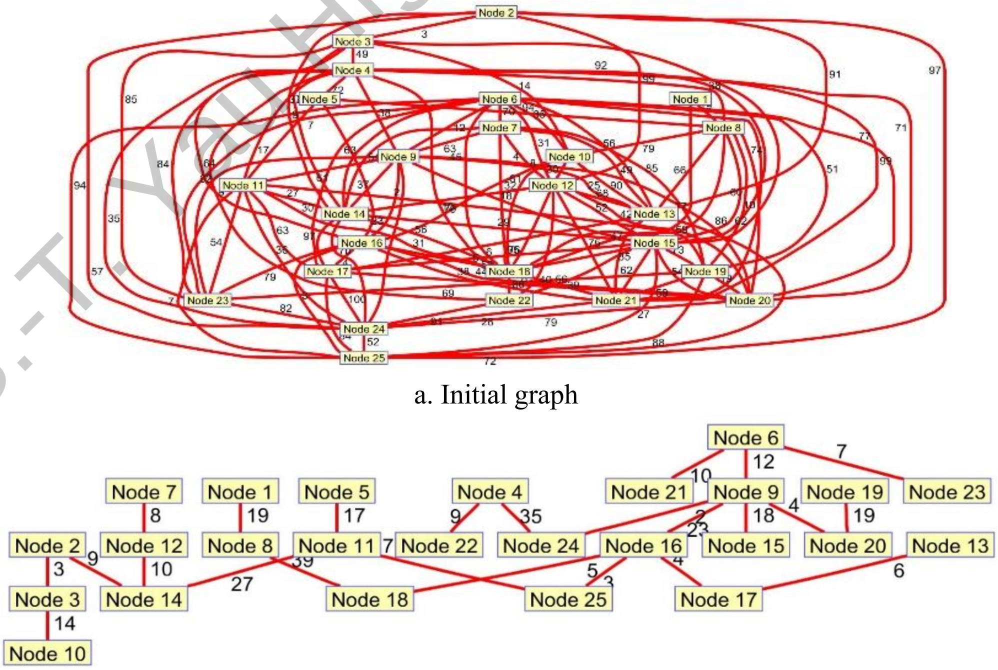
a. Initial network b. MST c. Clustering results

Fig. 7 Example 4: The number of edges in the network before and after optimization are 9 and 7, respectively. The total edge lengths before and after optimization are 42 and 32, respectively. The nodes are divided into two clusters.



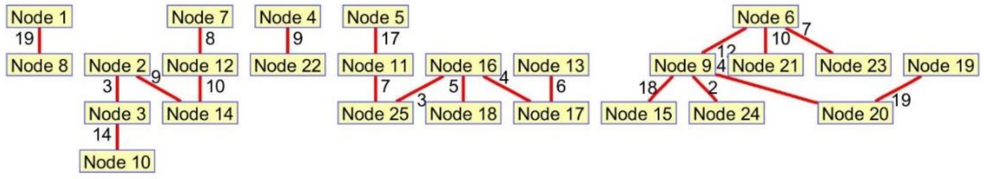
a. Initial network b. MST c. Clustering results

Fig. 8 Example 5: The number of edges of the network before and after optimization are 11 and 5, respectively. The total edge lengths before and after optimization are 539 and 111, respectively. The nodes are divided into two clusters.



a. Initial graph

b. MST



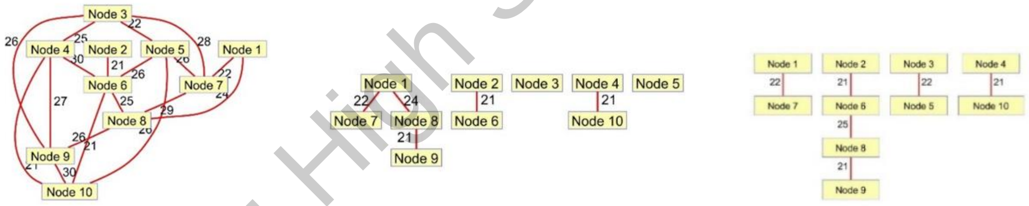
c. Clustering results

Fig. 9 Example 6: The number of edges of the network before and after optimization are 113 and 24, respectively. The total edge lengths before and after optimization are 5,879 and 310, respectively. The nodes are divided into five clusters.

5. Discussion

5.1 Keeping the Number of Graph Edges Sufficiently High and the Results Stable

When PMA-MST is used, the distribution of the edge lengths will affect the result. Sometimes, the number of edges in the shortest path network after optimization is less than n − 1. This occurs because the optimization is too strong and too many edges are removed, resulting in the failure of all nodes to be connected. Moreover, there may be multiple results, such as in the following example.



a. Original network b. Result 1: 5 edges c. Result 2: 6 edges

Fig. 10 Number of edges in the network before optimization are 18 and the numbers of edges in the optimized network are 5, and 6 for results 1 and 2, respectively, and the nodes cannot be connected to form a tree.

To avoid generating unreasonable optimization results, PMA-MST can be improved in three ways, as follows:

1. Normalize all edge lengths using

$$L'_k = \frac{L_k - \bar{L}}{\sigma} \tag{8}$$

Here, L_k is the length of each edge, $k = 1, 2, \dots m,$ where m is the total number of edges, $\bar{L} = \frac{\sum_{k=1}^{m} L_k}{m}$, $\bar{L}$ is the mean of the edges, and σ is the standard deviation, i.e., $\sigma = \sqrt{\frac{1}{n-1}\sum_{k=1}^{m}(L_k - \bar{L})^2}$.

However, because the edges calculated by Eq. (8) can generate negative values, the algorithm is inaccurate, and hence the result is increased by 1 so that all the edge lengths become positive, that is,

$$L''_k = L'_k + 1 \ . \tag{9}$$

2. Increase n_k, the number of randomly sown seeds

Increasing the number of random trials can make the results more stable and avoid obtaining different structures for the MST. (In many cases, the MST is not unique, and hence it is possible to search for different MSTs. The final total edge length can also be checked. As long as the total edge length after each optimization is nearly equal, the result is stable.) In theory, when more random trials are performed, it is more likely that the correct result (that is, the correct MST) will be obtained, and the result will be more stable. We generally randomly sow seeds i times the number of nodes n . For example, if the total number of nodes is n, the number of simulation times is $sk = n \times i$, where i is some factor.

3. Adjust the final flux of each edge

In the PMA-MST algorithm, after n_k simulation experiments, $Q_{ij}^{mean} = s_{ij}/nk$. If the number of edges of the generated graph is less than $n - 1$, Q_{ij}^{mean} is multiplied by amplification factor λ, that is,

$$Q_{ij}^{mean} = s_{ij}/nk \bullet \lambda. \tag{10}$$

This is then rounded to obtain Q_{ij}^{end}, where λ > 1. Because λ is greater than 1, it increases Q_{ij}^{mean}. Hence, the probability that Q_{ij}^{mean} is larger than 0 increases after rounding. As a result, the probability that edges are retained increases. For an edge to be retained, the algorithm that needs to be adjust is the original $Q_{ij}^{end} =$. We have $Q_{ij}^{end} \geq 1$ for the edges that are retained

After normalizing all edges in the above example and multiplying by amplification factor λ, the MST obtained by PMA-MST is shown in Fig. 11. The result indicates that all measures can improve the unconnected results caused by too few edges in the original algorithm. Moreover, the calculation result of the example is stable.

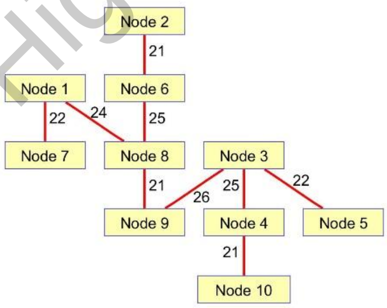


Fig. 11 Edges of the network are normalized, and the amplification factor λ is used to search for the MST. The total number of edges of the MST is 9 and the total length of edges is 207. Prim's method was also used to obtain the MST, and the results are the same.

5.2 Influence of β on Clustering

Parameter β is introduced in clustering to determine the grouping standard. For example, first, according to PMA-MST algorithm, the same tree is obtained, as shown in the Fig. 12. However, different values of β can result in different classification results. For instance, when $\beta = 0$, the nodes are divided into 10 clusters; when $\beta = 0.5$, 1, 2, and 3, the nodes are divided into eight clusters, five clusters, three clusters, and one cluster, respectively.

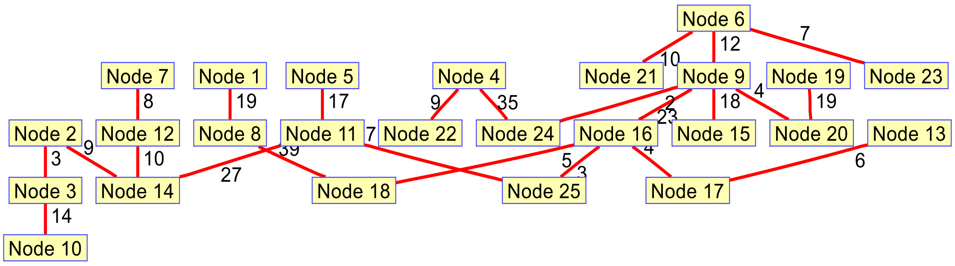


Fig. 12 MST of a network

When $\beta > 3$, the result is almost exactly the MST, and the nodes are no longer classified. If is β larger, fewer edges are cut and fewer groups are used for division. When β is smaller, more edges are cut and more groups are used for division.

5.3 Role of Acceleration Parameter α

The original function of acceleration parameter α is to make the algorithm converge faster, but in practical application, it is found that this parameter can control the structure of the generated MST. Acceleration parameter α has the ability to control the output of the optimal path graph.

For example, in the above example, there are 25 nodes, 114 edges, and the total edge length is 5,958. The results obtained when α changes while other parameters are fixed are listed in Table 2.

Further research into the role of α is a task for future work.

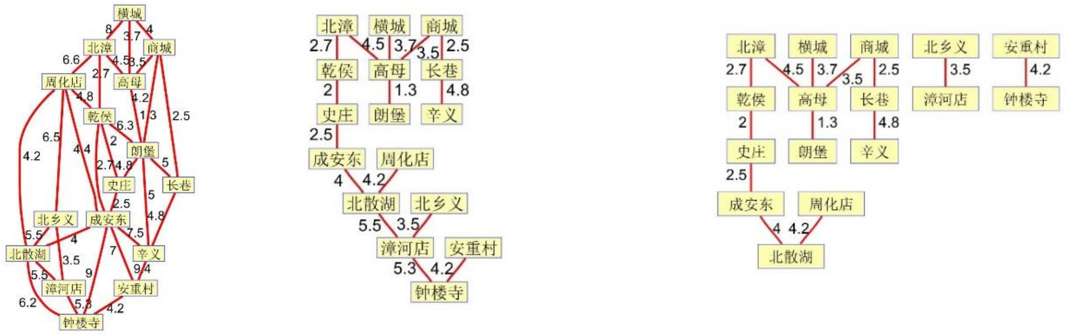
Table 2 Results of different values of acceleration parameter α, which can control the structure with the most efficient path

α	Optimized number of edges	Optimized the total edges length	a tree or not
2	24	322	No
5	24	319	Yes
7	24	319	Yes
10	19	172	No

6. Application

In this study, we consider the optimization of power grid: Fig. 13 shows the layout of transformer substations and power supply points, which consist of 16 power consumption points centered on townships and villages[14]. The aim is to optimize the regional power grid.

[14] https://www.docin.com/p-508584798



a. Original power grid diagram

b. Optimized power grid diagram

c. Partition of the power grid

Fig. 13 Power grid optimization. The number on each edge indicates the distance, and $\alpha = 7$, $\beta = 1$, and $\lambda = 1$. The total edge lengths of the network before and after optimization are 160.9 and 54.2, respectively. The numbers of edges in the graph before and after optimization are 33 and 15, respectively, and the graph is divided into three clusters.

The total edge length of MST in the original layout is 54.4, and the optimized result of PMA-MST is 54.2. The structure is also different, which indicates that the proposed method is better than the original method.

In the Iris dataset, there are 150 data samples that are divided into three categories, and each data contains four attributes. [15]Sample data are as follows:

Table 3 Sample from the Iris dataset

Id	Sepal Length Cm	Sepal Width Cm	Petal Length Cm	Petal Width Cm
1	5.1	3.5	1.4	0.2
2	4.9	3	1.4	0.2
……	……	……	……	……
150	5.9	3	5.1	5.9

Because each sample is a row, there are 150 rows in total, that is, 150 samples. The four attributes are four columns, and hence the data form a 150×4 matrix.

First, these data should be converted into a network. There are two ways to generated graphs: The first method uses Euclidean distance, that is, the distance between every two nodes (samples) is calculated by the following equation:

$$d_{i,j} = \sqrt[2]{\left(x_i^1 - x_j^1\right)^2 + \left(x_i^2 - x_j^2\right)^2 + \left(x_i^3 - x_j^3\right)^2 + \left(x_i^4 - x_j^4\right)^2}, \tag{11}$$

where *i,j* represents the number of nodes and there are four attributes in total. As an example, the distance between sample 1 and sample 2 is

$$d_{1,2} = \sqrt[2]{(5.1 - 4.9)^2 + (3.5 - 3)^2 + (1.4 - 1.4)^2 + (0.2 - 0.2)^2}.$$

Finally, a 150×150 matrix representing the distance between any two nodes is formed, with a total of 11,175 edges.

The second method uses the correlation coefficient[16], that is,

[15] https://baike.baidu.com/item/IRIS/4061453.

[16] There are many definitions of correlation coefficient. The Pearson correlation coefficient is used in this paper, which is a statistical index designed by statistician Karl Pearson. It is a quantity used to study the degree of linear correlation between variables, which is generally represented by *r*, which ranges between −1 and 1. An *r* closer to 0 indicates that the linear relationship is weaker.

$$r = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2 \cdot \sum_{i=1}^{n}(y_i-\bar{y})^2}} \quad . \tag{12}$$

However, in the second method, a larger absolute value of *r* indicates a greater similarity between two samples (vectors), and a smaller value means the similarity between two samples (vectors) is smaller. Therefore, after calculating *r*, we take the reciprocal of *r* $c_{ij}=1/r_{ij}$, and obtain a matrix *C* of 150×150 as the weight ("distance") between each node in the network.

The above two methods generate edges between any two nodes of the 150 samples in the Iris dataset, which is a complete graph[17]. There are hence $C_{150}^2 = 11{,}175$ edges.

Table 4 Result of clustering

	Method 1, $L_{ij}=d_{ij}$	Method 2, $L_{ij}= c_{ij}$
Parameter	α=10，β=2.2，λ=1	α=10，β=2.2，λ=1
Clustering results	8	3

In this example, eight network classes are formed using distance and three are formed when the correlation coefficient is used. The results of the correlation coefficient method seem to be closer to the classifications of biologists. However, different applications may yield different results. We believe that clustering is subjective, and its rationality should be evaluated when different answers are produced.

7. Conclusion

In this paper, a stochastic method was introduced into the PM algorithm of *P. polycephalum* and the PMA-MST algorithm was proposed. The PMA-MST algorithm was constructed to simulate the intelligent foraging behavior of *P. polycephalum* in a multi-source food environment. This algorithm can determine the MST of a network. In addition, to increase the applicability and efficiency of the algorithm, the PMA-MST algorithm includes acceleration coefficient *α* and amplification coefficient *λ*. These parameters accelerate the computational efficiency of the algorithm and facilitate the search for the MST. Given the MST, a threshold value is set according to the mean and standard deviation of the edge lengths, and edges exceeding this threshold value in the MST are removed to obtain a forest for clustering. The results of examples in this paper show that the proposed PMA-MST algorithm is an effective solution for searching for the MST and performing clustering. Because of the randomness embedded in the algorithm, its uncertainty leads to natural parallel characteristics of the algorithm, which is beneficial for improving the efficiency of the algorithm and determining the globally optimal solution. The algorithm was demonstrated to be better than the network in an original paper on power grid optimization. Moreover, the results on the test dataset proves that the data can be clustered reasonably.

In this paper, the generation process of swarm intelligence was explored based on computational principles. The extraordinary phenomena in nature continue to provide inspiration for the generation of intelligence, for which we are grateful.

[17] A complete graph is a simple undirected graph in which every pair of different vertices is connected by an edge.

Reference

（1）.　A. Adamatzky. Physarum machines Encapsulating reaction diffusion to compute spanning tree[J]. Naturwissenschaften, 2007,94(12):975-980

（2）.　A. Adamatzky. Physarum Machines: Computers from Slime Mould. Singapore. World Scientific Publishing Company[M], Illustrated edition (2010)

（3）.　A. Adamatzky. From reaction-diffusion to Physarum computing[J]. Natural Compution,2009,8(3):431~447

（4）.　A. Adamatzky., and J. Jones. Road planning with Slime Mold: if Physarum built motorways it would route M6/M74 through Newcastle. Int. J. Bifurcation Chaos [J]. 2010, 20(10):3065–3086

(5).　A. Schumann, K. Pancerz. Physarum Chip: Growing Computers from Slime Mould[J]. Logical Aspects. Publisher: Naukowe Wydawnictwo IVG.2016.9

（6）.　A . Tero, K. Yumiki, R.Kobayashi et al. Flow-network adaptation in Physarum amoebae[J]. Theory in Biosciences, 2008,127:89-94

（7）.　A. Tero, R. Kobayashi, and T. Nakagaki. A mathematical model for adaptive transport network in path finding by true slime mold[J]. Journal of theoretical biology. 2007,244(4):553-564,.

（8）.　A. Tero, R. Kobayashi, T. Nakagaki. Physarum solver: A biologically inspired method of road-network navigation [J]. Physica A: Statistical Mechanics and its Applications, 2006, 363(1): 115-119.

（9）.　A. Tero, S. Takagi, T. Saigusa, K. Ito, D. P. Bebber, M. D. Fricker, K. Yumiki, R. Kobayashi, and T. Nakagaki. Rules for biologically inspired adaptive network design[J]. Science, 2010,327, 5964, 439-442.

（10）.　C. Oettmeier，K. Brix，H. G. Döbereiner. Physarum polycephalum - A new take on a classic model system[J]. Journal of Physics D Applied Physics.2017.8, 50(41): 413001

（11）.　G Beni, J Wang. Proceedings of NATO Advanced Workshop on Robots and Biological Systems Tuscany, Italy, June 26–30, 1989:703

(12).　H. Zhou. Application of Physarum Model in Network System Construction and Reconfiguration [D].Southwest University, Master Thesis,2012

（13）.　I. Block, W. Briegleb, V. Sobick and K E. Wohlfarth-Bottermann. Confirmation of gravisensitivity in the slime mold Physarum polycephalum under near weightlessness[J]. Advances in Space Research. 1986, 6:143–50

（14）.　J. Jones. The emergence and dynamical evolution of complex transport networks from simple low-level behaviours[J]. International Journal of Unconventional computing. 2010,6:125-144

（15）.　J. MacQueen. Some Methods for Classification and Analysis of Multivariate Observations [C]. In: Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability. Berkeley, University of California Press, 1967: 281-297

（16）.　L. Kari, G. Rozenberg. The many facets of natural compu ting [J]. Commun. ACM, 2008, 51(10):72-83.

(17).　M. Kramar，K. Alim. Encoding memory in tube diameter hierarchy of living flow network[J]. PNAS, 2021,118 (10) :e2007815118

（18）.　M-A I Tsompanas, R Mayne, G C Sirakoulis and A Adamatzky. A cellular automata bioinspired algorithm designing data trees in wireless sensor networks[J]. Int. J. Distribute Sensor Networks . 2015, 6:1-10

（19）． N. Shadbolt. Nature-inspired computing [J] . IEEE Intelligent Systems, 2004, 19(1): 2-3.

（20）． P. Wang, J Li. Relation between Uncertainty and Parallelism of Natural Computation[J]. Chinese Journal of Computers, 2010, 37(2):216-220

（21）． Q. Liu, R. Zhang, Z. Zhao, Z. Wang, M. Jiao, G. Wang. Robust MST-Based Clustering Algorithm[J]. Neural Computation,2018, 30 (6): 1624–1646.

（22）． R. Jothi, S. K. Mohanty, & A. Ojha. Fast approximate minimum spanning tree based clustering algorithm[J]. Neurocomputing. 2018,272, 542–557

（23）． T. Nakagaki, H. Yamada, A. Toth. Maze-solving by an amoeboid organism[J]. Nature, 2000 ,407:470

（24）． V. Bonifaci. Andrew Adamatzky:Physarum Machines: Computers from Slime Mould [J]. Genetic Programming and Evolvable Machines, 2013, 14(1): 123-124.

（25）． Von Schweinitz L D. Synopsis Fungorum. Supplement Naturforschende Gesellschaft in Leipzig (Leipzig: Naturforschende Gesellschaft). 1822

（26）． Y. Nakagaki, H. Yamada, A. Toth. Intelligence: Maze-solving by an amoeboid organism [J]. Nature. 2000,407(6803):470

（27）． Y-P Gunji, T Shirakawa, T Niizato, M Yamachiyo, I Tani. An adaptive and robust biological network based on the vacant-particle transportation model[J]. Journal of Theoretical Biology. 2011,272(1):187-200

（28）． Y-P Gunji, T Shirakawa, T Niizato, T Haruna. Minimal model of a cell connecting amoebic motion and adaptive transport networks[J]. Journal of Theoretical Biology, 2008, 253(4): 659-667

致谢

“好奇心”往往被看作是一种优秀的品质，因为它往往会触发“求知欲”，从而自我驱动。例如，因为好奇“天空为什么是蓝色”而求问，回答这个问题会涉及光学、生物学以及神经科学的知识。当问题被科学自洽地解释清楚时，提问者和回答者都有一种满足的喜悦感。但是，有一些“好奇心”却会被认为是胡思乱想，比如“人类如何思考问题”。被提问者常常因为 “生命”、“意识”和“思维”这类带有过多终极意味的问题超出了已有的知识边界而感觉恼火。而我就是那个带有胡思乱想的“好奇心”的提问者。

生命的神奇之处往往蕴含在简单之中。2020 年的一天，我在网络上无意中看到了关于一种叫多头绒菌的生物的介绍。作为一种没有神经系统的“低等生物”，它竟然可以构造出类似东京地铁网络布局的结构，实在是太令人惊奇了。通过进一步学习和研究这种黏菌的各种“事迹”，发现很多科学家已经开始对它进行研究，包括计算机领域通过对多头绒菌的演化规则进行抽象，能够完成寻找路径的工作。我也尝试对多头绒菌的智能行为进行了研究，模拟它群体智能的工作方式。

或许只是“念念不忘，必有回响”。我在胡思乱想的“好奇心”驱动下，**在老师的指导下独立完成了这篇小论文。中山大学数学学院宋亮教授和华南师范大学附属中学的杨晓安老师在本文的定题、基本理论和写作要点等方面无偿地进行了悉心指导，对两位老师表示衷心的感谢！**

感谢我的父母一直鼓励我胡思乱想的“好奇心”!

文中引用了已有相关研究中的图片等，相关内容都在文中给出了来源。英文写作感谢 Kimberly Moravec 博士帮助对文字语法进行润色。

由于本人学识有限，难免错疏，敬请批评指正。