Network Community Detection Based on the *Physarum*-Inspired Computational Framework

Chao Gao, Mingxin Liang, Xianghua Li, Zili Zhang, Zhen Wang, and Zhili Zhou

Abstract—Community detection is a crucial and essential problem in the structure analytics of complex networks, which can help us understand and predict the characteristics and functions of complex networks. Many methods, ranging from the optimization-based algorithms to the heuristic-based algorithms, have been proposed for solving such a problem. Due to the inherent complexity of identifying network structure, how to design an effective algorithm with a higher accuracy and a lower computational cost still remains an open problem. Inspired by the computational capability and positive feedback mechanism in the wake of foraging process of *Physarum*, a kind of slime, a general *Physarum*-based computational framework for community detection is proposed in this paper. Based on the proposed framework, the inter-community edges can be identified from the intra-community edges in a network and the positive feedback of solving process in an algorithm can be further enhanced, which are used to improve the efficiency of original optimization-based and heuristic-based community detection algorithms, respectively. Some typical algorithms (e.g., genetic algorithm, ant colony optimization algorithm, and Markov clustering algorithm) and real-world datasets have been used to estimate the efficiency of our proposed computational framework. Experiments show that the algorithms optimized by *Physarum*-inspired computational framework perform better than the original ones, in terms of accuracy and computational cost.

Index Terms—Community detection, *Physarum*, genetic algorithm, ant colony optimization, Markov clustering algorithm

1 Introduction

Many complex systems in the real world can be organized and formulated as a network, such as internet [1], [2], social networks [3], biological networks and protein networks [4], [5]. Researchers in the field of network science have pointed that the structure analytics of complex networks provides a powerful and easy-to-implement technology for us to understand and predict the characteristics and functions of a network [6]. The community structure is one of the most basic and important structure features of a complex network [7], which plays an important role in the potential functional characteristics of a network (such as the small-world property [8], the scale-free property [9]), and the dynamical characteristics of a network (such as the cascading and epidemical diffusion in a network [10], [11]).

In a network with a community structure, vertexes across communities are connected sparsely, and vertexes within a community are connected densely [12]. The technology of community detection seeks to reveal relevant modules or communities at different granular levels that could affect or explain the global behavior of a whole complex system [13]. The community detection is used to detect the function of biology systems based on genomic data [14], predict the protein functions based on protein-protein interaction data [15], identify spreading behaviors based on social media [16], and uncover topics based on World Wide Web [17]. Due to its use in analyzing complex systems or networks, ranging from biology and engineering to social sciences and medicine, the technology of community detection has drawn widespread attention [18], [19].

Due to the importance of community structure, many algorithms have been proposed for detecting communities [2], [20]. Generally speaking, there are two typical kinds of algorithms for community detection: optimization-based and heuristic-based algorithms. Concerning the optimization-based algorithms, evolutionary algorithms have been recognized as the typical ones, which are inspired by principles from biology (e.g., biological genetics) and ethology (e.g., ant colony), and have been used for solving diverse kinds of optimization problems [21]. Generally, optimization-based algorithms formulate the problem of community detection as maximizing an objective function (e.g., modularity Q [6]) or multi-objective functions (e.g., kernel k-means and ratio cut [22]). Then, some evolutionary algorithms (e.g., Genetic Algorithm (GA) [23] and Ant Colony Optimization algorithm (ACO) [24], [25]) are used to solve such functions.

In term of heuristic-based algorithms, Markov clustering algorithms (e.g., R-MCL [20]) and their optimized algorithms based on the multi-level framework (e.g., MLR-MCL [26]) are representative ones, which simulate a dynamic process on networks for community division, rather than optimize specific objective function. During the Markov chain-based dynamic process, the flow simulation transfers in a network based on the expanding and contracting process alternately.

- *C. Gao, M.X. Liang, and X. Li are with the College of Computer and Information Science & College of Software, Southwest University, Chongqing 400715, China. E-mail: cgao@swu.edu.cn, 41809778@qq.com, li_xianghua@163.com.*
- *Z. Zhang is with the College of Computer and Information Science & College of Software, Southwest University, Chongqing 400715, China, and the School of Information Technology, Deakin University, Locked Bag 20000, Geelong VIC 3200, Australia. E-mail: zhangzl@swu.edu.cn.*
- *Z. Wang is with Northwestern Polytechnical University, Xi'an 710072, China. E-mail: nkzhenwang@163.com.*
- *Z. Zhou is with the College of Computer Science, Nanjing University of Information Science and Technology, Nanjing 210044, China. E-mail: zhou_zhili163net@163.com.*

Manuscript received 26 Sept. 2016; revised 6 Dec. 2016; accepted 7 Dec. 2016. Date of publication 13 Dec. 2016; date of current version 6 Dec. 2018. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TCBB.2016.2638824

With the transition process going on, the flow within a community will flow together. The state transition is implemented by an expansion operator, which spreads the flow out of a vertex to potential new vertices, and the enlarging inhomogeneity is implemented by an inflation operator, which strengthens the strong flows and weakens the weak flows. When the flows converge, the high-flow regions are clustered together [20].

Currently, lots of strategies have been proposed to release the potential of community detection algorithms, such as parallelization [27] and penalty strategy [28], or to extend the application to the field of overlapping community [29], [30] and directed community [31]. However, the key mechanism (e.g., the Markov chain-based dynamic process for enlarging the inhomogeneity), which plays an important role in the accuracy and computational cost of community detection, attracts few attentions based on our survey. In this paper, we aim to propose a bio-inspired computational framework to optimize the computational efficiency of both optimization-based and heuristic-based mining algorithms.

Biological studies have recently demonstrated the intelligence of a slime, *Physarum*, which shows an ability to solve maze problems and design networks without a central organ [32], [33]. The *Physarum* foraging behavior consists of two simultaneous self-organized processes: expansion and contraction [34]. Specifically, the features of morphological diversity and the trend growth of protoplasmic streaming are emerged based on the expanding process. Such features have been used for transport network designing [35] and route planning [36]. And the contracting process shows a positive feedback loop between the thickness of protoplasmic tubes and internal protoplasmic streaming. According to such features, many models and algorithms have been proposed to solve complex problems, such as the traveling salesman and 0/1 knapsack problems [37], and the bi-objective shortest path problem [38]. Among those models, a *Physarum*-inspired mathematical model (denoted as PM) has been proposed based on the feedback system in the contracting process [39], which shows an ability of accelerating convergence and improving the searching capability of evolutionary algorithms (e.g., GA [40] and ACO [37]) in terms of accuracy and computational cost.

Due to the aforementioned observations, we find that PM has a powerful computational capability, which has been used for optimizing some evolutionary algorithms and solving complex optimization problems. Meanwhile, we also observe that the flow simulations are similar to Markov clustering algorithms (e.g., R-MCL and MLR-MCL), both of which include expansion and contraction processes. However, the contraction process of Markov clustering algorithms mainly maps the flow to itself. In contrast, the contraction process of PM is implemented based on a feedback relationship between the pseudopodia and the protoplasmic fluxes. Therefore, we raise the following two questions.

- Does the *Physarum*-based mathematical model (PM) have a potential to recognize community structures in complex networks?
- Can we build a *Physarum*-inspired computational framework for optimizing optimization-based algorithms (e.g., GA and ACO) and heuristic-based algorithms (e.g., R-MCL and MLR-MCL) in terms of the accuracy and computational cost?

Considering the questions mentioned above, the main contributions of this paper are as follows.

1) Inspired by PM, a *Physarum* network mathematical model (PNM) is proposed, which has an ability to identify the inter-community edges from intra-community edges in a network. To the best of our knowledge, the ideas of PM have seldom been utilized to address the community detection. According to such computational capability, we aim to use PNM to optimize the initial phase of GA and ACO in order to improve the efficiency of community detection.
2) Using the feedback mechanism captured from the *Physarum* foraging [39], we modify the inflation operator in typical Markov clustering algorithms. Based on such a modified inflation operator, a novel positive feedback mechanism emerges, which could optimize Markov clustering algorithms in terms of accuracy and computational cost.

The remainder of this paper is organized as follows. Section 2 sheds light on the basic ideas of existing work to readers. Section 3 proposes a *Physarum*-based general computational framework for community detection. Then, some typical algorithms (e.g., GA, ACO, Markov clustering) are used for estimating the efficiency and scalability of proposed framework in Section 4. Finally, Section 5 concludes this paper.

2 Related Works

2.1 Algorithms for Community Detection

Given the practical significance of community detection, many algorithms and strategies have been proposed for detecting and revealing the community structure in networks. In general, those algorithms mainly fall into two categories: the optimization-based and heuristic-based algorithms.

2.1.1 Optimization-Based Community Detection

Concerning the perspective of optimization-based algorithms, a basic hypothesis for community detection is that the quality of community division can be evaluated by single or multiple objective functions. In general, those objective functions are based on networks topologies (e.g., the modularity Q [6]) or stochastic models (e.g., the likelihood function [41]). Some searching algorithms (e.g., evolutionary algorithms [18], [42], swarm optimization algorithms [43], and gradient descent methods [19]) are used to solve such objective functions. Based on the optimal solutions (i.e., the maximal values of objective functions), community structures can be obtained. In the following, two typical optimization algorithms (i.e., ACO and GA) are used to describe the process of community detection of such algorithms.

One of the typical optimization algorithms is the genetic algorithm. It is a computational model for simulating Darwin's theory of evolution by the natural selection and genetic mechanism. GACD [44] and GLAS [45] are two representative GA-based community detection algorithms. Each chromosome denotes a candidate community division of a network, and gene in the chromosome denotes the

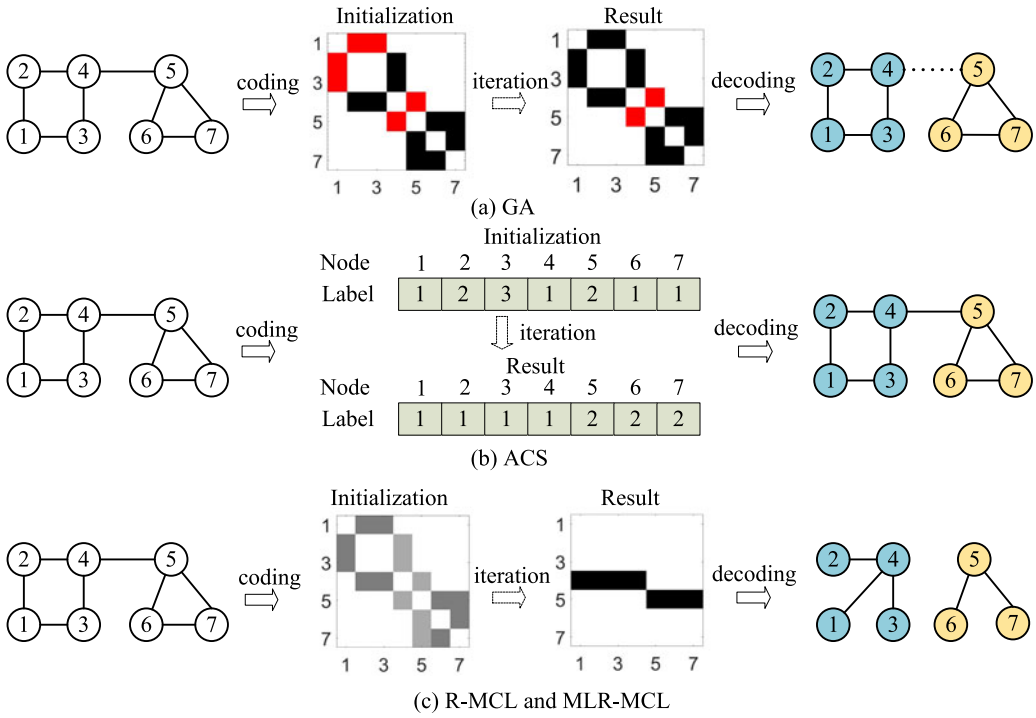


Fig. 1. The graphical illustration of different algorithms for community detection. A small synthetic network is used to report the coding and decoding processes of different algorithms: (a) The generic illustration of particle representation. A community division is coded as a matrix A, in which $A_{i,j}$ stands for the properties of corresponding edges. The red (i.e., $A_{i,j} = -1$) and black (i.e., $A_{i,j} = 1$) squares indicate the inter-community and intra-community edges, respectively. (b) The locus-based adjacency scheme for ACS. A community division is coded as a sequence (i.e., an ant), in which each number stands for the community label of corresponding node. (c) Coding and encoding processes of Markov clustering algorithms for community detection. A community division is coded as a matrix M, in which $M_{i,j}$ is between 0 and 1, and represented by the gray scales. More specifically, such value stands for the proportion of corresponding flows from node v_j to v_i.

community label of each node. Some objective function values (e.g., Q) are used to measure the fitness of chromosomes. Then, GA-based algorithms search chromosomes with the maximal fitness in each iteration based on the crossover and mutation operations. Fig. 1a illustrates the coding and decoding processes of GA-based community detection algorithm. Some strategies, such as NGACD in [23], have been proposed to improve the computational efficiency by optimizing the coding scheme. However, the accuracy and computational cost still limit its application.

The other typical optimization algorithm is ant colony optimization (ACO), which has two important elements during the optimization process. One is the pheromone matrix and the other is the heuristic factor [24], [25]. In each iteration, the pheromone matrix provides an indirect communication platform for ants and direct ants to find a better solution. After all ants finish the searching process, the pheromone matrix will be updated by the division results of ants. Fig. 1b illustrates the coding and decoding process of ACO-based community detection algorithms. Because the heuristic factor reflects the structural characteristics of a network, we can use such information to improve the quality of solution. Moreover, a global optimal solution can be derived when all of local solutions, found by each ant, have aggregated to one based on the clustering ensemble [43].

2.1.2 Heuristic-Based Community Detection

Heuristic algorithms detects communities based on intuitive assumptions or heuristic rules, rather than specific objective functions. The basic hypothesis is the dynamic process taking on a network, such as the Markov chains. And communities emerge in the wake of the dynamic process.

For example, the Markov clustering algorithm (e.g., R-MCL [20] and MLR-MCL [26]) is one of the most commonly used clustering algorithms in the field of bioinformatics. Its success is due to its easy implementation, high scalability, and strong robustness to the effect of topological noise [26]. The Markov clustering algorithms simulate a process of flow diffusion in such networks, in which every vertex has a quantity of fluxes, and the distribution of fluxes is represented by a matrix, called the flow matrix. Moreover, the flowing of fluxes is described by a Markov chain-based dynamic process with a transfer probability matrix, which can enlarge the inhomogeneity of flows based on the expansion and contraction process [20]. With the diffusion of such a stream, the flows in the networks can aggregate at certain vertices. During iterations, fluxes of vertexes in a tightly-linked group will flow together, and those vertexes are clustered as a community [46], as shown in Fig. 1c. Although the Markov clustering algorithms have been extended to solve overlapping communities [30] and directed [31] communities, the Markov chain-based dynamic process for enlarging the inhomogeneity isn't addressed in existing researches. This dynamic process is the key mechanism of Markov clustering algorithms and plays a crucial role in reconciling the efficiency of community detection in terms of accuracy and computational cost.

2.2 The Computational Features of *Physarum*

The plasmodium of the slime mould *Physarum polycephalum* is a large amoeba-like cell consisting of a dendritic network of tube-like pseudopodia. Recently, some studies have reported

that *Physarum* has shown abilities of solving maze problem and designing network in bio-experiments [32], [33]. The mechanism helping *Physarum* fulfil such missions without the centralized control, has drawn more and more attentions. Similar to the research of some nature-inspired computational methods (e.g., artificial immune system [47]), researchers pursue their studies in the *Physarum* from two different domains: the modeling-based analysis and *Physarum*-based computing.

In the field of the modeling-based analysis, lots of researchers have proposed a variety of models for characterizing and reappearing the foraging process of *Physarum*, which can help us better capture and reveal the core foraging features and mechanisms of *Physarum* [34], [48]. Based on these models, three basic features and mechanisms (i.e., trend growth [36], morphological diversity [49] and positive feedback loop [50]) have been extracted and characterized from the foraging process of *Physarum*.

Based on the above features, some *Physarum*-based computational frameworks have been proposed for solving hard complex problems, such as maze problem [51], network planning [52], transport network designing [53] and the shortest path problem [54]. Among those models, a *Physarum*-inspired mathematical model (denoted as PM) is one of the most effective methods because of its computational capability, which has been widely applied to optimize the nature-inspired algorithms. For example, taking advantage of PM, a universal optimization strategy is proposed to improve the efficiency and robustness of ant colony optimization [37], [55]. Moreover, the computational ability of PM has also been used to improve the efficiency of genetic algorithms and ant colony optimization algorithms, used for solving the multicast routing problem [40] and the bi-objective shortest path problem [38], respectively.

Inspired by analyses above, this paper aims to extend the application of PM to solve community detection. Two motivations encourage us to pursue such problem. *First*, we aim to integrate the computational capability of PM into the initialization of optimization-based algorithms (i.e., GA and ACO), in order to improve the performances of community detection problem. *Second*, considering that both PM and Markov clustering algorithms are based on the flow diffusions with the expansion and contraction processes in a network, we want to develop a novel Markov clustering algorithm by integrating the *Physarum*-inspired feedback system into Markov clustering algorithms (i.e., P-MLC and MLR-MCL). To the best of our knowledge, this is the first report that focuses on the techniques of community detection based on the computational capability of *Physarum*.

3 *Physarum* Computational Framework for Community Detection

In this section, Section 3.1 introduces the core computational mechanism of *Physarum*. Sections 3.2 and 3.3 formulate the *Physarum*-inspired computational framework for improving the efficiency of optimization-based and heuristic-based community detection algorithm, respectively.

3.1 The *Physarum*-Based Network Mathematic Model

Inspired by *Physarum*, Tero et al. have proposed a mathematical model denoted as PM [39], which has shown the abilities of path finding [39], network designing [33] and algorithms optimizing [37]. The model basically assumes an approximate Poiseuille flow of cytoplasmic fluxes from a source (denoted as s) to a destination (denoted as d) in the *Physarum* network [33]. There are two main processes for characterizing the protoplasmic streaming. *First*, with $D_{i,j}$ standing for the conductivity of an edge $e_{i,j}$ and p_i representing the pressure of a node v_i, the flux $Q_{i,j}$ can be expressed as shown in Eq. (1) based on the Poiseuille law. Assuming all vertices are zero capacity, the flux at each vertex can be expressed as shown in Eq. (2) by considering the conservation law at each time step. *Second*, ${Q^t}_{i,j}$ feeds back to ${D^t}_{i,j}$ based on Eq. (3), which indicates that ${D^t}_{i,j}$ changes over time according to the flux ${Q^t}_{i,j}$. Such feedback relationship between the cytoplasmic fluxes and conductivities of tubes is the core mechanism of PM, which ensures that the conductivities of tubes with lager fluxes are reinforced and that those with smaller fluxes degenerate. In the wake of such process, a high efficient network is emerged

$$ {Q^t}_{i,j} = \frac{{D^{t-1}}_{i,j}}{L_{i,j}} |{p^t}_i - {p^t}_j| \tag{1} $$

$$ \sum_i {Q^{t-1}}_{ij} = \begin{cases} I_0, \ if \ v_j \ is \ an \ inlet \\ I_0, \ if \ v_j \ is \ an \ oulet \\ 0, \ others \end{cases} \tag{2} $$

$$ {D^t}_{i,j} = \frac{Q^t_{i,j} + D^{t-1}_{i,j}}{k}. \tag{3} $$

In this paper, a *Physarum*-based network mathematical model (denoted as PNM) is designed to distinguish the inter-community edges from the intra-community ones in networks. Different from PM in which there is only one pair of inlet and outlet in each iteration, the major modification of PNM is the scheme of inlets/outlets choosing: when a vertex is chosen as an inlet in PNM, the others are chosen as outlets. In other words, there are many pairs of inlets and outlets during the iteration process. And each interaction of PNM consists of many interactions of PN. Therefore, through being combined with Eqs. (1) and (2) is modified as Eq. (4) where $|V|$ denotes the total number of vertexes in networks. Given a pair of certain inlet and outlet, we can construct a set of equations based on Eq. (4), in which D and L are known. By solving such equations set, the pressure of vertex v_i, denoted as p_i, can be calculated. When v_i is chosen as the inlet, a local conductivity matrix, denoted as $D^t(i)$, is calculated based on the feedback system. At the end of each iteration, the global conductivity matrix is updated by the average of local conductivity matrixes based on Eq. (5). Algorithm 1 shows a detailed description of PNM

$$ \sum_i \frac{{D^{t-1}}_{i,j}}{L_{i,j}} |{p^t}_i - {p^t}_j| = \begin{cases} I_0, \ if \ v_j \ is \ an \ inlet \\ \frac{-I_0}{|V|-1}, \ others \end{cases} \tag{4} $$

$$ D^t = \frac{1}{|V|} \sum_{i=1}^{|V|} D^t(i). \tag{5} $$

Algorithm 1. PNM for Community Detection

Input: An adjacent matrix A
Output: A conductivity matrix D
Initializing D^0 and the maximal iteration step T
For t from 1 to T
  For i from 1 to $|V|$
    Choosing v_i as the inlet
    Calculating $p^t{}_i$ based on Eq. (4)
    Calculating Q^t based on Eq. (1)
    Updating $D^t(i)$ based on Eq. (3)
  End for
  Updating D^t based on Eq. (5)
End for
Outputting D^T

3.2 *Physarum*-Inspired Initialization for Optimization-Based Algorithms

The priori knowledge is the basic component of evolutionary algorithms, which plays an important role in the computational efficiency [21]. According to different coding schemes and initialization methods, we aim to optimize GA-based and ACO-based algorithms through optimizing initial random candidate solutions and heuristic factors based on the computational capability of PNM, respectively. Fig. 2 shows the flow chart of optimizing process.

3.2.1 Physarum-*Inspired Initialization for GA-Based Community Detection Algorithms*

This section takes NGACD [23], a typical GA-based community detection algorithm, as an example to introduce the $Physarum$-inspired initialization scheme. The initial candidate solution for community detection is optimized through distinguishing the inter-community edges from the intra-community ones based on the computational capability of PNM. A matrix DA is used to denote the property of $e_{i,j}$, in which $da_{i,j} = 1$ if and only if $e_{i,j}$ is an intra-community edges. More details are summarized as follows.

Algorithm 2. The Initialization of GA Based on PNM

Input: An adjacent matrix A
Output: An initial community division DA
Initializing $DA{=}{-}A$
Choosing a set of vertexes IC as central vertexes
For every vertex v_{ic} in IC
  For every edge $e_{ic,j}$
    If $e_{ic,j} \notin$ the top R% conductivities D in Algorithm 1 **Then** $DA_{ic,j} = -DA_{ic,j}$
  End for
End for

First, a conductivity matrix D is obtained based on Algorithm 1. And then, all edges are supposed to be inter-community, i.e., $DA = -A$. Thereafter, some vertexes are chosen randomly to be central points. Meanwhile, the edges joining those chosen vertexes are marked as intra-community edges (i.e., $da_{i,j} = 1$), except for the edges with the top conductivities. After that, DA could be used to generate an initial solution (i.e., the preliminary community divisions). At the end of initialization phase, we need to repeat the above process for several times until getting the preestablished amount of initial solutions. The whole process is shown in Algorithm 2. Based on DA, the vertexes connected by intra-communities edges are identified as a community, and the optimized initialization of GA for community detection is completed. After that, the work is moving on to code them into chromosomes.

Algorithm 3. The Framework of P-NGACD

Input: An adjacent matrix A
Output: Final community division
Calculating the conductivity matrix D based on Algorithm 1
Initializing DA based on Alg. 2 where R=10$\sim$ 20 percent
Sorting the fitness value of population based on Q
While not satisfy the terminal condition **do**
  Executing the crossover operation based on fitness
  Executing the mutation operation
  Calculating the Q values of chromosomes
  Forming a new generation based on the Q values
End while

For a clear expression, a prefix (i.e., P-) is added to the original names of optimized algorithms for distinguishing. In P-NGACD, a division is coded as a matrix $genome_{N\times N}$, where $N = |V|$. $genome_{i,j}$ describes the attribute partition of edge between vertexes i and j. Similar to the definition of DA, $genome_{i,j} = 1$ denotes that two vertexes belong to the same community. Otherwise, $genome_{i,j} = -1$. Besides, if there is no edge between two vertexes, $genome_{i,j} = 0$. Based on such coding scheme, it is convenient for crossover operation and mutation operation by exchanging the attribute of edges and changing a gene bit respectively. The detailed process of P-NGACD is shown in Algorithm 3.

3.2.2 Physarum-*Inspired Initialization for ACO-Based Community Detection Algorithms*

This section takes ACOC [25], a typical ACO-based community detection algorithm, as an example to introduce how PNM can be used to optimize the heuristic factors of ACO. Such strategy can also optimize other ant colony clustering algorithms, such as IACONet [24]. For differing the original algorithms, the optimized algorithms are marked with a prefix (i.e., P-). We first provide the formulation of original ACO-based community detection algorithms. And then, we highlight the major optimization process based on PNM.

As discussed in Section 2.1.1, the pheromone matrix (i.e., $Tau(i, c_j)$) and heuristic factor (i.e., $\eta(i, c_j)$) are two basic elements of ACO. Based on the coding scheme of ACOC [25], each ant aims to detect a community division based on a probability matrix directly in each iteration. Such probability matrix is denoted as $P(i, c_j)$, which indicates the probability of vertex i belonging to a community C_j. And c_j stands for the label of community C_j

$$P(i, c_j) = \frac{\eta(i, c_j)^{\beta} Tau(i, c_j)^{\alpha}}{\sum_k \eta(i, c_k)^{\beta} Tau(i, c_k)^{\alpha}}. \tag{6}$$

The $\eta(i, c_j)$ in Eq. (6) is a heuristic factor, which helps improve the search capability of ants based on the adjacent matrix of networks. For example, such factor in ACOC

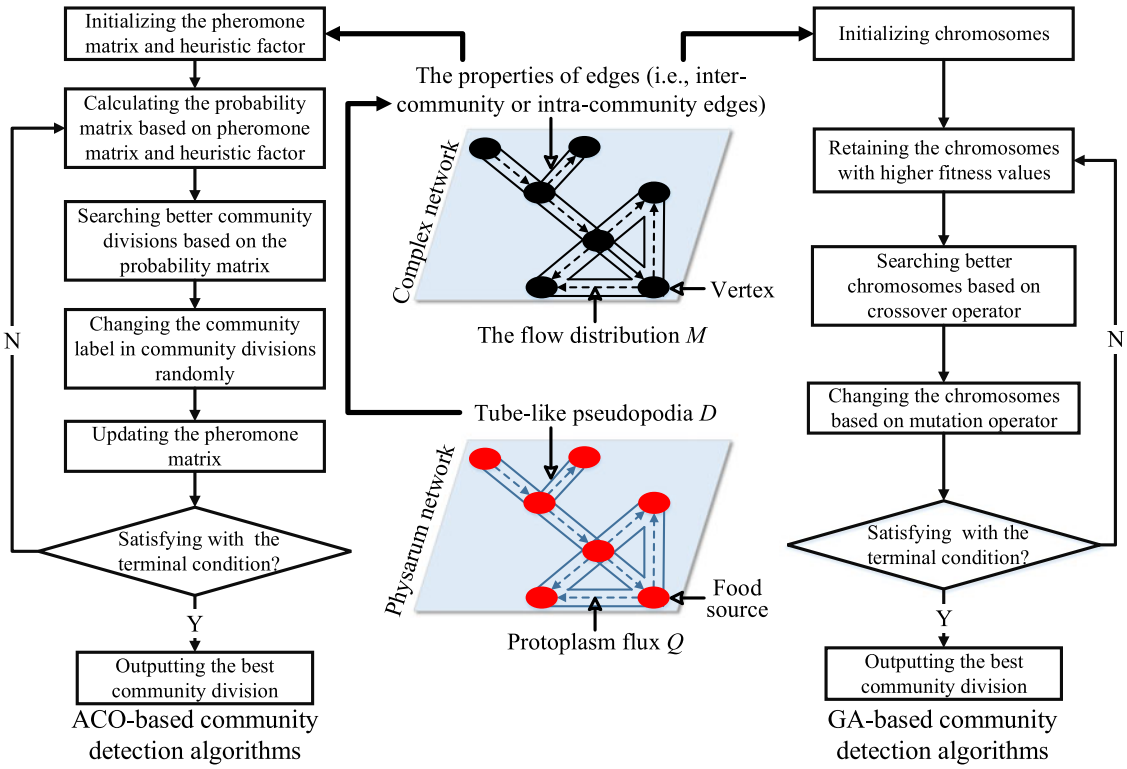


Fig. 2. The flow chart of *Physarum*-inspired initialization for optimization-based algorithms. The priori knowledge, provided by PNM, is used to optimize the initial candidate solutions of GA and the heuristic factor of ACO, respectively.

denotes the number of edges connecting the vertexes in the community C_j from vertex i. Based on the structural characteristic of a community, the more vertexes in community C_j vertex i connects, the larger probability of belonging to the community C_j it will have. Therefore, the $\eta(i, c_j)$ of ACOC is computed based on Eq. (7) in which n_{i,c_j} indicates the number of edges connecting vertex i and vertexes in community C_j. And Tau is updated by the ants in each iteration based on the qualities of solutions

$$\eta(i, c_j) = n_{i,c_j} \Big/ \sum_{k=1}^{p} n_{i,c_k}. \tag{7}$$

This paper optimizes the heuristic factor based on PNM as shown in Eq. (8), in which $n^*{}_{i,c_j} = \sum_{k \in C_j} \frac{1}{w_{ik}}$. And w_{ik} indicates the conductivity of e_{ik} which is an element in the matrix D outputted by Algorithm 1. c_j stands for the label of community C_j. With such expression, $\eta^*(i, c_j)$ has a larger value when there are more intra-community edges connecting vertex i and vertexes in the community C_j, compared with the original $\eta(i, c_j)$, vice versa. The optimized heuristic factors exaggerate the inhomogeneity of original heuristic factor, and offer more obvious information to ants, which improve the search capability of original algorithms

$$\eta^*(i, c_j) = n^*{}_{i,c_j} \Big/ \sum_{k=1}^{p} n^*{}_{i,c_k}. \tag{8}$$

Based on the matrix $P(i, c_j)$ in each iteration, the *local strategy* of ACOC is used to search a community label. For example, the vertex i has a probability p_0 to be assigned a community label c_j with the maximal P value (i.e., $arg\ \max_j P(i, c_j)$), and has a probability $1 - p_0$ to be assigned a random label based on the roulette way. After searching a community label, ants plan to *update the pheromone matrix Tau*. There are two phases: The first phase is implemented when an ant finishes its searching in an iteration. In this phase, every ant updates the pheromone matrix based on Eq. (9), in which ρ is a parameter of evaporation rate, and Q indicates the quality of community division the ant finds. The second phase is executed after all the ants finish their local searching. Based on the community divisions with the top Q values in the current iteration, Eq. (10) is implemented for enhancing the effects of better community divisions. More specifically, the *random walk strategy* is a kind of random searching process in ACOC, which aims to improve the diversity of solutions and protect ant colony clustering algorithms from premature. In the adopted random walk strategy, each vertex is reassigned by a random community label with a probability P_m. And the solution is accepted, if and only if the reassignment improves the modularity value of corresponding solution

$$Tau(i, c_j) = (1 - \rho) \cdot Tau(i, c_j) + 2\rho \cdot Q \cdot \delta(i, c_j) \tag{9}$$

$$Tau(i, c_j) = (1 - \rho) \cdot Tau(i, c_j) + Q_{top} \cdot \delta(i, c_j). \tag{10}$$

With the local searching and updating for Tau, ants will aggregate to some community divisions with higher Q values. And the solution with the highest Q value will be outputted as the optimal community division. The detailed steps are reported in Algorithm 4.

Algorithm 4. *Physarum*-Inspired ACO for Community Detection

Input: Adjacent matrix A, the number of communities NC
Output: The community division of a network
1: Initializing parameters including $\alpha, \beta, p_0, p_m, t$
2: Endowing edges with weights D based on Algorithm 1
3: Initializing the Tau and η matrixes
4: **While** $t <$ maximal iteration
5: **For** each ant
6: Computing the P matrix based on Eq. (6)
7: Searching a community division based on the matrix P
8: Updating the pheromone matrix based on Eq. (9)
9: **End for**
10: Executing the random walk strategy for all divisions
11: Updating the pheromone matrix based on Eq. (10)
12: **End while**
13: Outputting the division with the maximal Q value.

3.3 *Physarum*-Inspired Feedback Loop for Heuristic-Based Algorithms

As a typical heuristic-based algorithms, Markov clustering algorithms have been used for community detection. However, the Markov chain-based dynamic process attracts little attention. Inspired by the positive feedback relationship between the cytoplasmic fluxes and tube conductivities of $Physarum$ network, this section aims to build a relationship between expansion and inflation operators in order to improve the efficiency of the Markov clustering algorithms.

3.3.1 Formulation of Community Detection Based on MCL

The expansion and contraction processes are the main processes of MCL [20], which are implemented based on two matrices (i.e., the flow distribution matrix M and canonical transition matrix T) and three operators (i.e., expansion, inflation, and pruning operators). Specifically, an element $m_{i,j}$ in M represents the flux flowing to v_i from v_j, and an element $t_{i,j}$ in T denotes the quantity of flux flowing to v_j from v_i in an iteration step. The initial M and T are derived from the adjacency matrix A of a network, based on

$$M^0_{i,j} = T^0_{i,j} = \frac{A_{i,j}}{\sum_k A_{k,j}}. \tag{11}$$

R-MCL is a typical Markov clustering algorithm, in which the expansion, inflation, and pruning operators are executed alternately. The expansion operator is used for spreading fluxes in a network. Such operator is conveniently implemented through multiplying the matrix by the canonical transition matrix T, as shown in Eq. (12). In order to prevent M from converging to the principal eigenvector of T, a dynamic process is imported for enlarging the differentiated values based on the inflation operator. Such operator raises every entry in M to the power of r and then normalizes the columns as shown in Eq. (13). As all of the entries in M are less than or equal to 1, the inhomogeneity in each column is enlarged (i.e., the strong flows are strengthen and the weak flows are weaken). By removing entries under a pre-established threshold, the pruning operator accelerates convergence and reduces non-zero entries to save memory. The thresholds are based on the average and maximum values within each column [20]

$$M^{st+1} = M^{st} \times T \tag{12}$$

$$M^{ts+1} = [M^{ts+1}_{i,j}]^r \Big/ \sum\nolimits_k [M^{ts+1}_{k,j}]^r. \tag{13}$$

By the end of iterations, the most of vertices will find an attractor, to which all of their fluxes flow. The vertices, whose fluxes flow to the same attractor, will be clustered to a community. Therefore, there is only one positive value in each column of M whose row indexes denotes the community labels of the corresponding vertices of the columns.

3.3.2 The Physarum-Inspired Markov Clustering Algorithm

As shown in Eqs. (1), (2), and (3), there exists a positive feedback relationship between the flux $Q_{i,j}$ and the conductivity of tube $D_{i,j}$, which reinforces the conductivities of tubes with lager fluxes and degenerates those with smaller fluxes. Because $D_{i,j}$ changes over time according to the flux $Q_{i,j}$, we discretize Eq. (3) to Eq. (14). As the total flux is kept constant (i.e., I_0), there is a competition between the edges

$$\frac{D^{ts+1}_{i,j} - D^{ts}_{i,j}}{\triangle ts} = |Q^{ts}_{i,j}|^u - D^{ts}_{i,j}. \tag{14}$$

Inspired by such feedback system, we aim to build a relationship between expansion and contraction processes in R-MCL by simulating the interaction between the fluxes and conductivities in PNM. The new algorithm is denoted as P-MCL through optimizing the dynamic process in R-MCL in terms of the multi-step expansion operator, the *Physarum*-inspired inflation operator, and the terminal condition.

During the multi-step expansion operator, P-MCL takes the effects of the flow length in each iteration step into consider in order to solve the problem of the excessive partition of R-MCL [20]. A l-length flow can be expressed by Eq. (15), and a larger network requires a larger l to control the number of communities

$$M^{st+1} = M^{st} \times \prod_{i=1}^{l} PT. \tag{15}$$

In the *Physarum*-inspired inflation operator, a feedback step is added in the wake of original inflation mechanism (i.e., Eq. (13)). To differ the notation of canonical transition matrix T in R-MCL, the *Physarum*-based transition matrix is denoted as PT. And inspired by Eq. (3) in PM, the flow distribution matrix M has an effect on PT based on Eq. (16). We further discretize the Eq. (16) to Eq. (17) for simplifying calculation, where λ stands for the time step of discretization and u is inherited from Eq. (3) in PM. And such two parameters collectively control the effect of M on PT. Because the sum of each column should still equal to 1 for a transition matrix. After the feedback step, a normalized step is implemented based on Eq. (18). PT then feeds back to M based on Eq. (12) in the next iteration step

$$\frac{d}{dt}PT_{i,j} = |M_{i,j}|^u - PT_{i,j} \tag{16}$$

$$PT_{i,j}^{ts+1} = (1-\lambda)\cdot[M_{i,j}^{ts}]^u + \lambda \cdot PT_{i,j}^{ts} \tag{17}$$

$$PT_{i,j}^{st+1} = PT_{i,j}^{st+1} \Big/ \sum\nolimits_k PT_{k,j}^{st+1}. \tag{18}$$

The terminal condition is measured by $Energy$ defined in Eq. (19), which uses the maximal value of changes among all of the entries in M to estimate the change of the whole M. If $Energy$ is less than a threshold, the system will stop

$$Energy^{ts} = Max\{|M_{i,j}^{ts} - M_{i,j}^{ts-1}| \ |\forall j,i\}. \tag{19}$$

Algorithm 5. *Physarum*-Inspired Markov Clustering Algorithm (P-MCL)

Input: A network adjacent matrix A
Output: A division result
1: A=A+I
2: Initializing M and PT based on Eq. (11)
3: Implementing flows based on Eq. (15)
4: Computing M^{ts+1} based on Eq. (13)
5: Updating the PT^{ts+1} based on Eqs. (17) and (18)
6: Deleting the elements in M^{ts}, which is smaller than the threshold obtained based on the pruning scheme
7: **If** the terminal condition is satisfied, go to Step **8**
Else go to Step **3**
8: Outputting the division result

The pseudo-code of P-MCL is represented in Algorithm 5. With an interaction between the flow distribution matrix M and *Physarum*-based transition matrix PT, a new feedback flow is constructed, which helps to improve the accuracy and reduce the computational cost of the algorithms. As a multilevel framework for obtaining large gains in speed is proposed in [20], we also adopt this multilevel framework to P-MCL, denoted as MLP-MCL. More detailed information about the multilevel framework can be found in [20].

4 Experiments

4.1 Datasets and Measurements

Some real-world networks collected by Newman,[1] Batagelj and Mrvar[2] are used to evaluate the accuracy and computational cost of our proposed framework. These networks have different scales and domains, including social networks (e.g., G_1) and electricity networks (e.g., G_{10}). The basic information of those networks is shown in Table 1.

There are two popular metrics for measuring the performance of community detection: the modularity Q and normalized mutual information (NMI). For a given network with N vertexes and E edges, $Q = \frac{1}{2*E}\sum_{i,j}(A_{i,j} - \frac{d_i d_j}{2*E})\delta(i,j)$ in which $\delta(i,j)=1$ or 0 mean that two vertexes are clustered into the same or different community. $A_{i,j}$ is an element in the adjacent matrix A, in which $A_{i,j}=1$ if and only if there is an edge connecting v_i and v_j. d_i denotes the degree of a vertex i. And $NMI(S,R) = \frac{-2\sum_{i=1}^{p_S}\sum_{j=1}^{p_R} Co_{ij}\cdot\log(\frac{Co_{ij}\cdot N}{C_i\cdot C_j})}{\sum_{i=1}^{p_S} Co_i\cdot\log(\frac{Co_i}{N}) + \sum_{i=j}^{p_R} Co_j\cdot\log(\frac{Co_j}{N})}$, in which S and R represent the standard community division and detected community division respectively. Co is the confusion matrix of S and R, in which $Co_{i,j}$ is the number of vertexes belonging to r_i of R and s_j of S at the same time. Co_i and Co_j indicate the sums of elements within row i and column j of Co, respectively. Thereby, if the standard community division is known, the quality metrics function could be $NMI(S,R)$ as well.

1. http://www-personal.umich.edu/ mejn/netdata/
2. http://vlado.fmf.uni-lj.si/vlado/vladonet.htm

TABLE 1
The Basic Features of Used Networks

No	Name	Vertex	Edges	k	C	$\#C$
G_1	KarateClub	34	78	4.588	0.588	2
G_2	Dolphins	62	160	5.129	0.303	2
G_3	PolBooks	105	441	8.400	0.488	3
G_4	Football	115	613	10.660	0.403	12
G_5	Lesmis	77	254	6.597	0.056	-
G_6	Adjnoun	112	425	7.589	0.190	-
G_7	Celegans	297	1,540	9.656	0.326	-
G_8	Roget	674	613	1.819	0	-
G_9	Netscience	1,589	2,742	3.451	0.878	-
G_{10}	PowerGrid	4,941	6,594	2.669	0.107	-

No represents the used notations of networks. k and C stand for the average degree and clustering coefficient. #C is the number of communities in standard community divisions, in which "-" means that standard division is nonexistent.

For a clear expression, a prefix (i.e., P-) is added to the original names of optimized algorithms for distinguishing. For example, the original GA-based community detection algorithm (i.e., NGACD [23]), ACO-based community detection algorithms (i.e., ACOC [25] and IACONet [24]), and Markov cluster algorithms (i.e., R-MCL [20] and MLR-MCL [26]) are denoted as P-NGACD, P-ACOC, P-IACONet, P-MCL and MLP-MCL, respectively. All experiments are implemented with the same parameters setting and running environment. Moreover, for some random algorithms (i.e., GA- and ACO-based algorithms), the results are averaged over 100 repeated runnings for eliminating fluctuation and evaluating the robustness of algorithms.

4.2 Accuracy Comparison

4.2.1 *The Accuracy of* Physarum*-Inspired Optimization-Based Algorithms*

In terms of GA-based community detection algorithms, we take NGACD [23] as an example to estimate the efficiency of our proposed framework. Fig. 3 plots the box chart of results returned by NGACD and P-NGACD. Due to the randomness of maximum and minimum, efficiency comparison is mainly based on the quartiles and means. Results show that the first and third quartiles, median and mean values of Q returned by P-NGACD are higher than those of NGACD in all datasets, which means that P-NGACD has a stronger exploring ability. Moreover, the lengths of boxes of P-NGACD are shorter than those of NGACD, which verify that the robustness of P-NGACD is stronger than that of NGACD. The visualization results of two algorithms, as shown in Fig. 4, show that P-NGACD can detect the community structures more exactly.

Taking the football network as an example, the dynamic averages and variances of Q are reported in Fig. 5. As is shown in Fig. 5a, the initial Q of P-NGACD is obviously higher than that of NGACD. Although the gap between

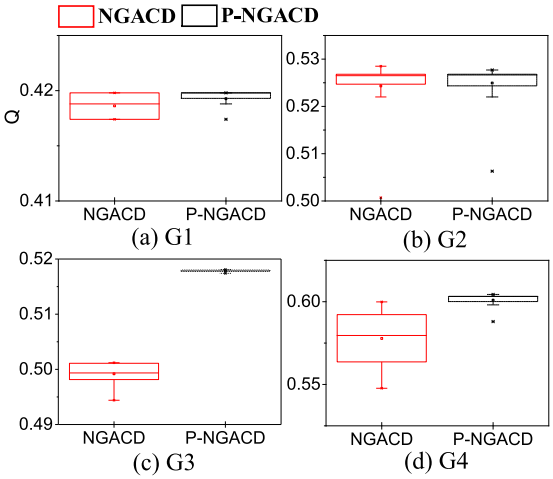


Fig. 3. Box charts of results returned by P-NGACD and NGACD in four networks in term of Q, in which the bottom and top of box are the first and third quartiles respectively, and the band inside the box denotes the median. The ends of whiskers represent the minimum and maximum of Q. Moreover, the small quadrates in boxes stand for the means of Q.

them is decreasing with the increment of generation, the final Q of P-NGACD also has a significant advantage. Moreover, the variances of P-NGACD keep a lower values with a small fluctuation than those of NGACD as shown in Fig. 5b. Those results indicate the proposed framework can enhance both the search capabilities and robustness of algorithms. In addition, the proposed framework can accelerate the convergence by comparing the iterations of two algorithms.

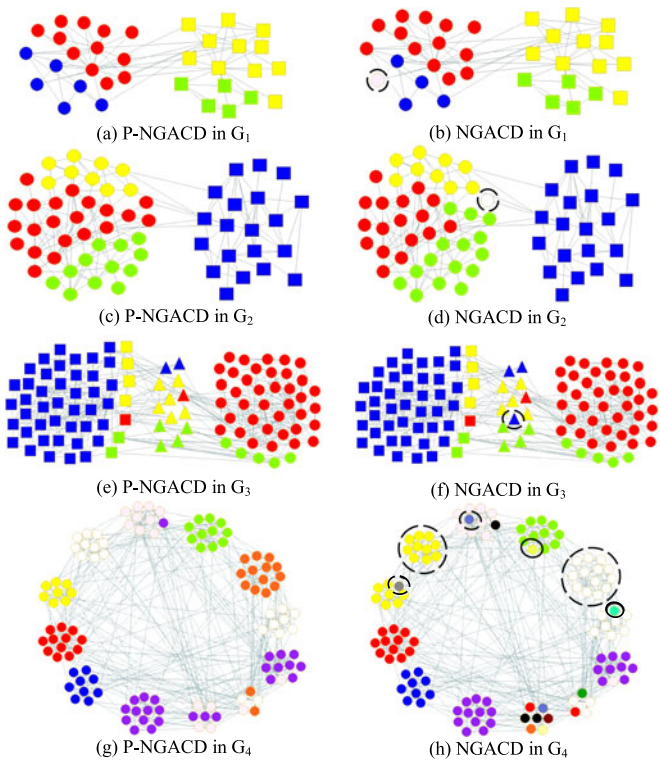


Fig. 4. Division results of P-NGACD and NGACD based on the maximal Q in (a) and (b) KarateClub network, (c) and (d) Dolphins network, (e) and (f) PolBooks network, and (g) and (h) Football network. The shapes of the vertices represent the known community structures, and the colors indicate the division results of the algorithms. The circle emphasizes the main difference in the results returned by NGACD and P-NGACD. The result of P-NGACD is more accurate than that of NGACD.

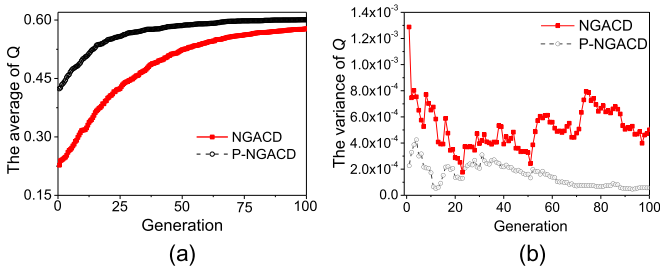


Fig. 5. Dynamic averages and variances of Q with the increment of generation in P-NGACD and NGACD in G_4 (i.e., Football network).

Concerning the ACO-based algorithms, Fig. 6 shows the box charts of modularity Q returned by ACOC[25], IACONet [24] and their optimized algorithms (i.e., P-ACOC, P-IACONet) in four real-world networks. The results indicate that the P-ACOC and P-IACONet have higher average values than those of ACOC and IACONet, respectively. And the first and third quartiles of optimized algorithms are also higher than those of original algorithms. Meanwhile, the distribution ranges of optimized algorithms are smaller, compared with original ones, which means the proposed strategy can enhance the robustness of ant colony algorithms.

Moreover, Fig. 7 reports the dynamic changes of averaged Q with the increments of iterations, in which the optimized algorithms have a higher growth rate than the original algorithms. And more especially, we can observe a distinct performance improvement at the end of iterations in terms of the modularity Q.

4.2.2 *The Accuracy of* Physarum*-Inspired Heuristic-Based Algorithms*

In this section, R-MCL [20] and MLR-MCL [26] are selected as typical heuristic-based algorithms for estimating the efficiency of our proposed framework. Different from experiments in the optimization-based algorithms, in which the quality of a solution can be estimated by the optimal modularity Q, the other metric NMI as discussed in Section 4.1 is used for measuring the similarity between detected communities and standard communities for heuristic-based community detection algorithms.

To estimate the performance of P-MCL, we compare our algorithm with other algorithms: the stochastic-model-based algorithm (shortened as Karrer) [41], PageRank-based algorithm (shortened as PPC) [56], and some novel algorithms (e.g., MNDP [19] and Combo [57]). The NMI values of those algorithms are reported in Table 2. The results indicate that P-MCL and MLP-MCL have the highest NMI values among the algorithms in datasets with different network scales, which means that P-MCL and MLP-MCL are the most accurate algorithms among them.

For the datasets with unknown communities, Table 3 reports the modularity Q values of algorithms in different datasets. The results show that P-MCL and MLP-MCL have the highest Q values in all of the networks except for G_{10}, in which P-MCL still has the third highest Q value. In order to demonstrate the differences of solutions quality and provide more convincing conclusion, the statistical comparison is implemented based on the methodology reported in [58]. Based on the rankings of algorithms in each dataset, as shown in Tables 2 and 3, we can infer that there are significant differences among the results of algorithms with a confidence of

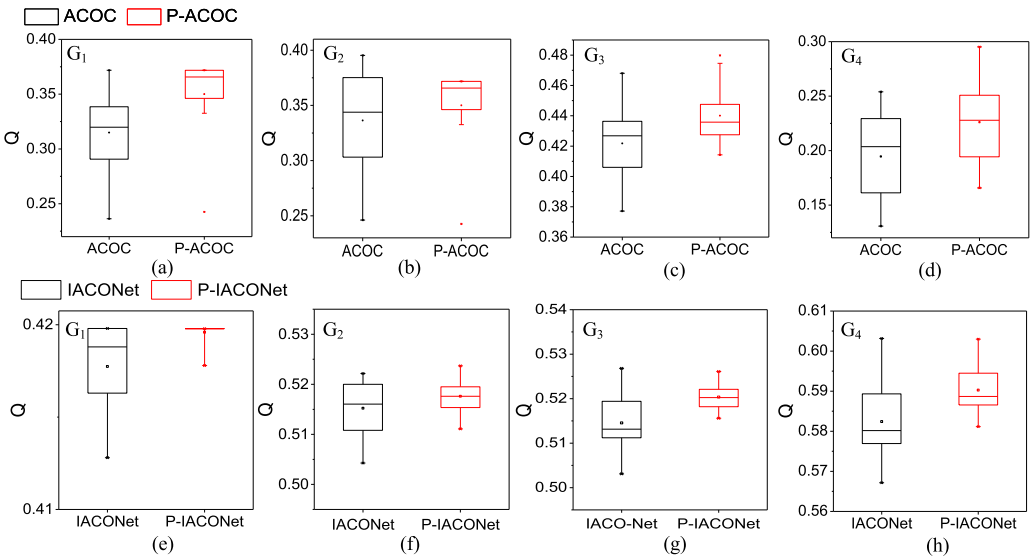


Fig. 6. Box charts of results returned by ACOC and P-ACOC in term of Q in (a) and (e) KarateClub, (b) and (f) Dolphins, (c) and (g) PolBooks, and (d) and (h) Football network, in which the bottom and top of the box are the first and third quartiles, and the band inside the box is the median. The ends of the whiskers represent the minimum and maximum of Q. Moreover, the small quadrates in boxes stand for the means of Q.

95 percent based on the Friedman test. Furthermore, we can conclude that the optimized algorithms are better than other algorithms with 95 percent confidence through executing the Holm's and Hochberg's procedures. All above results verify that the proposed updating algorithms could improve the accuracy of the Markov clustering algorithms and obtain clearer community structures.

4.3 Computation Complexity Analysis

In addition to accuracy, computational cost is also an important property of algorithm. In this section, the analysis of computational cost includes two parts: (1) theoretical analysis, denoted by big-O notation, and (2) comparison analysis, including running time in seconds and the coverage rate measured by the maximal iteration step (i.e., T). Moreover, the major modification for MCL is to improve the feedback system. Therefore, the statistical report on *Energy* changes in each step is used to measure and demonstrate the efficiency of the feedback system.

4.3.1 Theoretical Analysis of Time Complexity

(1) Physarum-Inspired Optimization-Based Algorithms. At each iteration, every vertex should be chosen as the inlet once. When a vertex is chosen, a corresponding system of equations needs to be solved. Therefore, there are N equations to be solved, and the worst computation complexity is O(N^3). With an empirical setting (i.e., $T = 1$), the total computation complexity of *Physarum*-inspired optimization is O(N^4), which is acceptable for a NP-complete problem.

(2) Physarum-Inspired Heuristic-Based Algorithms. To address different operations, the whole time complexity of P-MCL consists of three parts. The time complexity of the expansion operator is $O(\sum_{i=1}^{N} k^2)$ in which k is the average number of non-zero elements in each column (i.e., the average degree of a network). The time complexities of the inflation and pruning operators are both $O(E)$, in which E is the total number of edges in a network. As $E < N^2$, the total worst time complexity of P-MCL is presented in Eq. (20), where T is the maximal iteration step that is the key factor in the complexities of the Markov clustering algorithms. However, it is difficult to estimate T universally. A comparison of T is reported in Section 4.3.2, which can prove that the feedback system can reduce T significantly

$$O(T \times (N * k^2 + E + E)) < O(T \times N(k^2 + 2 * N)). \quad (20)$$

4.3.2 The Comparison Analysis of Time Complexity

Table 4 reports the running time of optimization-based algorithms and their optimized algorithms. All experiments are implemented on Matlab 2015b with an Intel i5-4460 CPU and 7.86 GB of RAM. Results demonstrate the *Physarum*-inspired optimized framework does not increase the computational complexity noticeably.

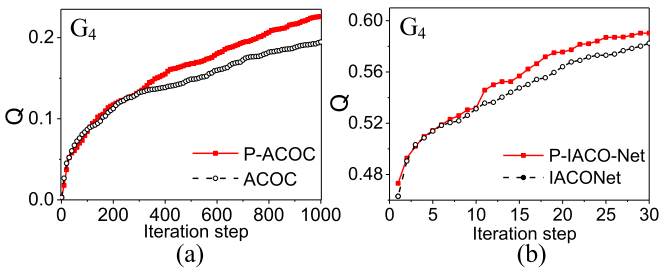


Fig. 7. The dynamic changes of averaged Q with the increment of iteration in the Football network. (a) is the compared results of ACOC and P-ACOC and (b) is the compared results of IACONet and P-IACONet. Results show that the proposed framework can accelerate the convergence rate of original algorithms.

TABLE 2
Comparison of Community Detection in Four Networks with Known Community Structures

ALGNET	G_1	G_2	G_3	G_4
Combo	68.73	57.15	56.03	89.03
MNDP	100.00	88.88	53.01	92.42
Karre	83.72	88.88	54.20	87.06
PPC	70.71	57.92	57.30	85.61
R-MCL	83.65	64.70	52.50	80.99
MLR-MCL	83.65	64.70	52.50	80.99
P-MCL	**100.00**	**88.88**	**59.23**	**92.83**
MLP-MCL	**100.00**	**88.88**	56.86	92.42

The results show that MLP-MCL and P-MCL have better NMI values and exhibit a significant improvement compared with MLR-MCL and R-MCL.

TABLE 3
Comparison Results in Real Networks with Unknown Community Structures in Terms of the Modularity Q

ALGNET	G_5	G_6	G_7	G_8	G_9	G_{10}
Combo	0.560	0.302	0.555	0.936	0.959	**0.939**
MNDP	0.543	0.271	-	-	0.833	0.868
Karre	0.457	−0.104	0.249	0.008	0.640	0.179
PPC	0.454	0.255	0.374	0.933	0.779	0.930
R-MCL	0.465	−0.112	0.128	0.924	0.962	0.814
MLR-MCL	0.465	−0.112	0.128	0.946	0.880	0.773
P-MCL	0.615	**0.376**	0.496	**0.956**	**0.971**	0.913
MLP-MCL	**0.654**	0.367	**0.555**	0.952	0.966	0.902

The greater the modularity Q value, the better the community structure obtained. The results show that MLP-MCL and P-MCL have better modularity values and exhibit a significant improvement compared with R-MCL and MLR-MCL.

TABLE 4
The Running Time of Optimization-Based Algorithms in Seconds

Alg. Net.	G_1	G_1	G_3	G_4
NGACD	6.9098	9.5531	14.9710	16.8733
P-NGACD	6.9708	9.5983	15.0266	17.0304
ACOC	2.1862	4.2678	8.4494	9.8882
P-ACOC	2.2814	4.3298	8.5971	9.9121
IACONet	1.0875	2.2906	7.2562	14.8844
P-IACONet	1.1906	2.4156	7.5031	15.4844

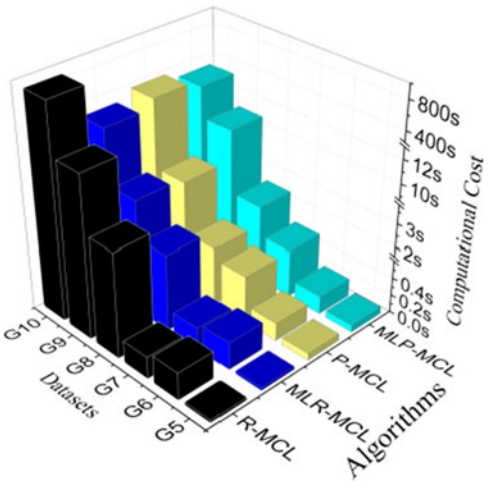


Fig. 8. Running time of R-MCL, MLR-MCL, P-MCL, and MLP-MCL in six networks with different scales. The results show that P-MCL and MLP-MCL have lower running times than R-MCL and MLR-MCL.

Fig. 8 plots the running time of algorithms in six networks with different network scales. According to this figure, P-MCL and PML-MCL have the lowest computational costs in four datasets (i.e., G_5, G_6, G_8, and G_9). For the other two datasets (i.e., G_7 and G_{10}), the differences are not obvious. After that, as the running environment has a significant effect on the running time of the algorithms, Fig. 9 reports the dynamic $Energies$ of R-MCL and P-MCL during the increments of iteration steps. Results show that $Energies_{P-MCL}$ drop faster than that of R-MCL, which are not affected by the running environments.

The statistical frequencies of $\Delta Energy^{ts} = |Energy^{ts} - Energy^{ts-1}|$ is reported in Fig. 10. Although PNM has

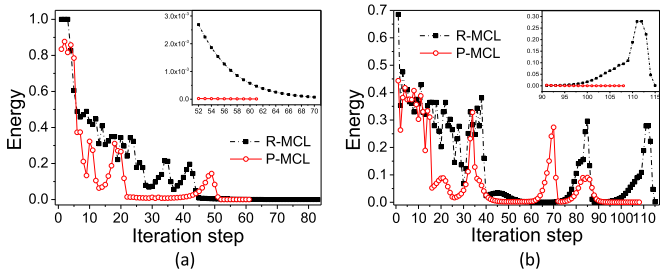


Fig. 9. Dynamic $Energies$ of P-MCL and R-MCL in (a) G_3 and (b) G_9. As shown in (a) and (b), $Energy_{P-MCL}$ has a faster descent rate and a smaller fluctuation than $Energy_{R-MCL}$, which results in a lower maximal iteration step.

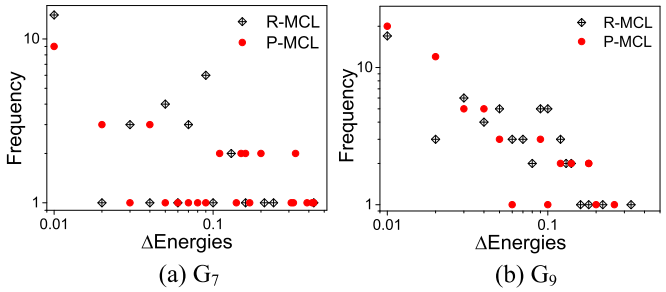


Fig. 10. Statistical frequencies of corresponding $\Delta Energy$ (logarithmic scale) in two networks with different scales. The results verify that P-MCL has a stronger feedback loop which can help P-MCL reduce computational cost and accelerate convergence rate.

different effects on R-MCL in G_7 and G_9 as shown in Fig. 8, P-MCL has more chances to obtain a larger $\Delta Energies$ in two networks, which indicates the P-MCL can obtain more profits at each iteration step, and accelerate mining process. Therefore, we can conclude that P-MCL has a stronger feedback loop than R-MCL, which leads to the reduction in time complexity and improvement of the convergence rate.

5 Conclusion

Inspired by the computational capability and positive feedback evolving process in the wake of foraging process of *Physarum*, we propose a modified *Physarum* network model (PNM) with a specific scheme for community detection, which identifies the inter-community edges from the intra-community edges and shows a positive feedback problem solving process. Taking the advantages of PNM, a *Physarum*-inspired computational framework is proposed to improve the efficiency of optimization-based and heuristic-based algorithms in terms of accuracy and computational cost.

First, according to the different coding schemes and initialization methods, the GA-based (e.g., NGACD) and ACO-based (e.g., ACOC and IACONet) community detection algorithms are optimized through optimizing initial random candidate solutions and heuristic factors respectively based on the *Physarum*-inspired priori knowledge. Second, referencing to the relationship between the cytoplasmic fluxes and tube conductivities of *Physarum* network, the dynamic processes of heuristic-based algorithm (e.g., R-MCL) and its multi-level framework algorithm (i.e., MLR-MCL) are optimized for enlarging the inhomogeneity through building a feedback relationship between the expansion operator and inflation operator in the Markov clustering algorithms.

We finally give demonstrative applications of the framework in a set of real-world networks to verify the performance

of *Physarum*-inspired optimization-based algorithms (i.e., P-NGACD, P-ACOC, P-IACONet) and *Physarum*-inspired heuristic-based algorithms (i.e., P-MCL, MLP-MCL). Experiments show that optimized algorithms with proposed framework have a higher accuracy and stronger robustness. Moreover, a computational complexity analysis verifies the scalability of our framework. In the future, we will further estimate the efficiency of our proposed framework in an open and dynamic environment, such as in the wireless mesh networks or underwater sensor networks [59], [60], [61]. More importantly, we should rethink our work in terms of distributed computing in order to meet the requirement of cloud computing [62], [63].

Acknowledgments

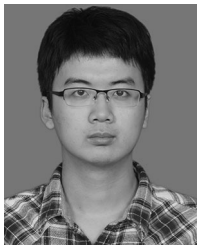
Prof. Zili Zhang and Dr. Xianghua Li are the corresponding authors. This work is supported by the National Natural Science Foundation of China (Nos. 61402379, 61403315), CQ CSTC (No. cstc2015gjhz40002), Fundamental Research Funds for the Central Universities (Nos. XDJK2016A008, XDJK2016B029) and Chongqing Graduate Student Research Innovation Project (No. CYS16067).

References

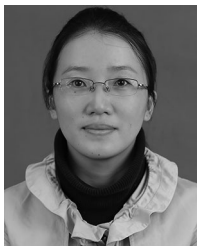
[1] S. H. Strogatz, "Exploring complex networks," *Nature*, vol. 410, no. 6825, pp. 268–276, 2001.
[2] N. Tremblay and P. Borgnat, "Graph wavelets for multiscale community mining," *IEEE Trans. Signal Process.*, vol. 62, no. 20, pp. 5227–5239, Oct. 2014.
[3] C. C. Lin, J. R. Kang, and J.-Y. Chen, "An integer programming approach and visual analysis for detecting hierarchical community structures in social networks," *Inf. Sci.*, vol. 299, pp. 296–311, 2015.
[4] Y.-Y. Ahn, J. P. Bagrow, and S. Lehmann, "Link communities reveal multiscale complexity in networks," *Nature*, vol. 466, no. 7307, pp. 761–764, 2010.
[5] S. S. Bhowmick and B. S. Seah, "Clustering and summarizing protein-protein interaction networks: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 3, pp. 638–658, Mar. 2016.
[6] M. Girvan and M. E. Newman, "Community structure in social and biological networks," *Proc. Nat. Academy Sci. USA*, vol. 99, no. 12, pp. 7821–7826, 2002.
[7] Q. Cai, M. G. Gong, S. S. Ruan, Q. G. Miao, and H. F. Du, "Network structural balance based on evolutionary multiobjective optimization: A two-step approach," *IEEE Trans. Evol. Comput.*, vol. 19, no. 6, pp. 903–916, Dec. 2015.
[8] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.
[9] A.-L. Barabasi and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.
[10] C. Gao, J. Liu, and N. Zhong, "Network immunization and virus propagation in email networks: Experimental evaluation and analysis," *Knowl. Inf. Syst.*, vol. 27, no. 2, pp. 253–279, 2011.
[11] A. Nematzadeh, E. Ferrara, A. Flammini, and Y. Y. Ahn, "Optimal network modularity for information diffusion," *Phys. Rev. Lett.*, vol. 113, 2014, Art. no. 088701.
[12] S. Wang, M. Gong, B. Shen, Z. Wang, Q. Cai, and L. Jiao, "Deep community detection based on memetic algorithm," in *Proc. IEEE Congr. Evol. Comput.*, 2015, pp. 648–655.
[13] M. T. Schaub, J.-C. Delvenne, S. N. Yaliraki, and M. Barahona, "Markov dynamics as a zooming lens for multiscale community detection: Non clique-like communities and the field-of-view limit," *PLoS ONE*, vol. 7, no. 2, 2012, Art. no. e32210.
[14] M. J. Herrgård, et al., "A consensus yeast metabolic network reconstruction obtained from a community approach to systems biology," *Nature Biotechnology*, vol. 26, no. 10, pp. 1155–1160, 2008.
[15] J. Lee, S. P. Gross, and J. Lee, "Improved network community structure improves function prediction," *Sci. Rep.*, vol. 3, 2013, Art. no. 2197.
[16] D. Centola, "The spread of behavior in an online social network experiment," *Science*, vol. 329, no. 5996, pp. 1194–1197, 2010.
[17] G. W. Flake, S. Lawrence, C. L. Giles, and F. M. Coetzee, "Self-organization and identification of web communities," *Computer*, vol. 35, no. 3, pp. 66–70, 2002.
[18] M. Gong, L. Ma, Q. Zhang, and L. Jiao, "Community detection in networks by using multiobjective evolutionary algorithm with decomposition," *Physica A: Statistical Mech. Appl.*, vol. 391, no. 15, pp. 4050–4060, 2012.
[19] D. Jin, Z. Chen, D. X. He, and W. X. Zhang, "Modeling with node degree preservation can accurately find communities," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 160–167.
[20] V. Satuluri and S. Parthasarathy, "Scalable graph clustering using stochastic flows: Applications to community discovery," in *Proc. 15th Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 737–746.
[21] M. Crepinsek, S.-H. Liu, and M. Mernik, "Exploration and exploitation in evolutionary algorithms: A survey," *ACM Comput. Surveys*, vol. 45, no. 3, pp. 1–33, 2013.
[22] M. Gong, Q. Cai, X. Chen, and L. Ma, "Complex network clustering by multiobjective discrete particle swarm optimization based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 18, no. 1, pp. 82–97, Feb. 2014.
[23] X. Li, C. Gao, and R. Pu, "A community clustering algorithm based on genetic algorithm with novel coding scheme," in *Proc. 10th Int. Conf. Natural Comput.*, 2014, pp. 486–491.
[24] C. Mu, J. Zhang, and L. Jiao, "An intelligent ant colony optimization for community detection in complex networks," in *Proc. IEEE Congr. Evol. Comput.*, 2014, pp. 700–706.
[25] P. Shelokar, V. Jayaraman, and B. Kulkarni, "An ant colony approach for clustering," *Analytica Chimica Acta*, vol. 509, no. 2, pp. 187–195, 2004.
[26] S. Fortunato, "Community detection in graphs," *Phys. Rep.*, vol. 486, no. 3, pp. 75–174, 2010.
[27] Q. Niu, P. W. Lai, S. M. Faisal, S. Parthasarathy, and P. Sadayappan, "A fast implementation of MLR-MCL algorithm on multi-core processors," in *Proc. 21st Int. Conf. High Perform. Comput.*, 2014, pp. 1–10.
[28] V. Satuluri, S. Parthasarathy, and D. Ucar, "Markov clustering of protein interaction networks with improved balance and scalability," in *Proc. 1st ACM Int. Conf. Bioinf. Comput. Biol.*, 2010, pp. 247–256.
[29] D. Jin, H. C. Wang, J. W. Dang, D. X. He, and W. X. Zhang, "Detect overlapping communities via ranking node popularities," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016, pp. 172–178.
[30] Y.-K. Shih and S. Parthasarathy, "Identifying functional modules in interaction networks through overlapping Markov clustering," *Bioinf.*, vol. 28, no. 18, pp. i473–i479, 2012.
[31] Y.-K. Shih, et al., "Component detection in directed networks," in *Proc. 23rd ACM Int. Conf. Inf. Knowl. Manage.*, 2014, pp. 1729–1738.
[32] T. Nakagaki, H. Yamada, and Á. Tóth, "Intelligence: Maze-solving by an amoeboid organism," *Nature*, vol. 407, no. 6803, 2000, Art. no. 470.
[33] A. Tero, et al., "Rules for biologically inspired adaptive network design," *Science*, vol. 327, no. 5964, pp. 439–442, 2010.
[34] Y. Liu, et al., "A new multi-agent system to simulate the foraging behaviors of physarum," *Natural Comput.*, pp. 1–15, 2015, [Online]. Available: http://link.springer.com/article/10.1007/s11047-015-9530-5
[35] Y. X. Liu, C. Gao, M. X. Liang, L. Tao, and Z. L. Zhang, "A physarum-inspired vacant-particle model with shrinkage for transport network design," in *Proc. 6th Int. Conf. Swarm Intell.*, 2015, pp. 74–81.
[36] J. Jones, "Applications of multi-agent slime mould computing," *Int. J. Parallel Emergent Distrib. Syst.*, vol. 32, no. 5, pp. 420–449, 2016.
[37] Y. Liu, et al., "Solving NP-hard problems with physarum-based ant colony system," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, 2015, [Online]. Available: https://www.computer.org/csdl/trans/tb/preprint/07172477-abs.html
[38] Z. L. Zhang, C. Gao, Y. X. Lu, Y. X. Liu, and M. X. Liang, "Multi-objective ant colony optimization based on the physarum-inspired mathematical model for bi-objective traveling salesman problems," *PLoS ONE*, vol. 11, no. 1, 2016, Art. no. e0146709.
[39] A. Tero, R. Kobayashi, and T. Nakagaki, "A mathematical model for adaptive transport network in path finding by true slime mold," *J. Theoretical Biol.*, vol. 244, no. 4, pp. 553–564, 2007.
[40] M. Liang, C. Gao, and Z. Zhang, "A new genetic algorithm based on modified physarum network model for bandwidth-delay constrained least-cost multicast routing," *Natural Comput.*, pp. 1–14, 2016, [Online]. Available: http://link.springer.com/article/10.1007/s11047-016-9545-6
[41] B. Karrer and M. E. Newman, "Stochastic blockmodels and community structure in networks," *Phys. Rev. E*, vol. 83, no. 1, 2011, Art. no. 016107.

[42] Z. Li and J. Liu, "A multi-agent genetic algorithm for community detection in complex networks," *Physica A: Statistical Mech. Appl.*, vol. 449, pp. 336–347, 2016.
[43] D. Jin, D. Liu, B. Yang, J. Liu, and D. He, "Ant colony optimization with a new random walk model for community detection in complex networks," *Advances Complex Syst.*, vol. 14, no. 5, pp. 795–815, 2011.
[44] M. Tasgin and H. Bingol, "Community detection in complex networks using genetic algorithm," *arXiv: 0711.0491*, 2007, [Online]. Available: https://arxiv.org/abs/cond-mat/0604419
[45] D. Jin, D. X. He, D. Y. Liu, and C. Baquero, "Genetic algorithm with local search for community mining in complex networks," in *Proc. 22nd Annu. IEEE Int. Conf. Tools Artif. Intell.*, 2010, pp. 105–112.
[46] S. Dongen, "Graph clustering by flow simulation," PhD dissertation, Centre for Mathematics and Computer Science, Universiteit Utrecht, Utrecht, Netherlands, 2001, [Online]. Available: http://micans.org/mcl/
[47] X. H. Li, Z. X. Wang, T. Y. Lu, and X. J. Che, "Modelling immune system: Principles, models, analysis and perspectives," *J. Bionic Eng.*, vol. 6, no. 1, pp. 77–85, 2009.
[48] Y. H. Wu, Z. L. Zhang, Y. Deng, H. Zhou, and T. Qian, "A new model to imitate the foraging behavior of physarum polycephalum on a nutrient-poor substrate," *Neurocomputing*, vol. 148, pp. 63–69, 2015.
[49] Y.-P. Gunji, T. Shirakawa, T. Niizato, and T. Haruna, "Minimal model of a cell connecting amoebic motion and adaptive transport networks," *J. Theoretical Biol.*, vol. 253, no. 4, pp. 659–667, 2008.
[50] A. Tero, R. Kobayashi, and T. Nakagaki, "A coupled-oscillator model with a conservation law for the rhythmic amoeboid movements of plasmodial slime molds," *Physica D: Nonlinear Phenomena*, vol. 205, no. 1–4, pp. 125–135, 2005.
[51] A. Adamatzky, "Slime mold solves maze in one pass, assisted by gradient of chemo-attractants," *IEEE Trans. Nanobiosci.*, vol. 11, no. 2, pp. 131–134, Jun. 2012.
[52] M. A. I. Tsompanas, G. C. Sirakoulis, and A. I. Adamatzky, "Evolving transport networks with cellular automata models inspired by slime mould," *IEEE Trans. Cybern.*, vol. 45, no. 9, pp. 1887–1899, Sep. 2015.
[53] D. P. Bebber, J. Hynes, P. R. Darrah, L. Boddy, and M. D. Fricker, "Biological solutions to transport network design," *Proc. Royal Soc. London B: Biol. Sci.*, vol. 274, no. 1623, pp. 2307–2315, 2007.
[54] J. Jones, "Characteristics of pattern formation and evolution in approximations of physarum transport networks," *Artif. Life*, vol. 16, no. 2, pp. 127–153, 2010.
[55] Z. Zhang, C. Gao, Y. Liu, and T. Qian, "A universal optimization strategy for ant colony optimization algorithms based on the physarum-inspired mathematical model," *Bioinspiration Biomimetics*, vol. 9, no. 3, 2014, Art. no. 036006.
[56] S. A. Tabrizi, A. Shakery, M. Asadpour, M. Abbasi, and M. A. Tavallaie, "Personalized PageRank clustering: A graph clustering algorithm based on random walks," *Physica A: Statistical Mech. Appl.*, vol. 392, no. 22, pp. 5772–5785, 2013.
[57] S. Sobolevsky, R. Campari, A. Belyi, and C. Ratti, "General optimization technique for high-quality community detection in complex networks," *Phys. Rev. E*, vol. 90, no. 1, 2014, Art. no. 012811.
[58] S. Garcla, D. Molina, M. Lozano, and F. Herrera, "A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: A case study on the CEC' 2005 special session on real parameter optimization," *J. Heuristics*, vol. 15, no. 6, pp. 617–644, 2009.
[59] S. D. Xie and Y. X. Wang, "Construction of tree network with limited delivery latency in homogeneous wireless sensor networks," *Wireless Pers. Commun.*, vol. 78, no. 12, pp. 231–246, 2014.
[60] J. Shen, H. W. Tan, J. Wang, J. W. Wang, and S. Y. Lee, "A novel routing protocol providing good transmission reliability in underwater sensor networks," *J. Internet Technol.*, vol. 16, no. 1, pp. 171–178, 2015.
[61] P. Guo, J. Wang, B. Li, and S. Y. Lee, "A variable threshold-value authentication architecture for wireless mesh networks," *J. Internet Technol.*, vol. 15, no. 6, pp. 929–936, 2014.
[62] Q. Liu, W. D. Cai, J. Shen, Z. J. Fu, X. D. Liu, and N. Linge, "A speculative approach to spatial-temporal efficiency with multi-objective optimization in a heterogeneous cloud environment," *Secur. Commun. Netw.*, vol. 9, no. 17, pp. 4002–4012, 2016.
[63] Y. Kong, M. J. Zhang, and D. Y. Ye, "A belief propagation-based method for task allocation in open and dynamic cloud environments," *Knowl.-Based Syst.*, vol. 115, pp. 123–132, 2016.

Chao Gao received the PhD degree in computer science from the International WIC Institute, Beijing University of Technology, in 2010. He is an associate professor in the College of Computer and Information Science, Southwest University, Chongqing, China. He has been a postdoctoral research fellow in the Computer Science Department, Hong Kong Baptist University. His main research interests include data-driven complex systems modeling, complex social networks analysis, and nature-inspired computing.



Mingxin Liang received the bachelor's degree in information and computing science from the Beijing Institute of Technology, Beijing, China, in 2014. He is currently working toward the postgraduate degree at Southwest University, Chongqing, China, majoring in computer application. His research interests include bio-inspired algorithm and intelligence algorithm.



Xianghua Li received the PhD degree in computer science from Jilin University, in 2009. She is an assistant professor in the College of Computer and Information Science, Southwest University, Chongqing, China. Her main research interests include complex network analysis, community detection, and nature-inspired computing.

Zili Zhang received the BSc degree in computing from Sichuan University, the MEng degree in computing from the Harbin Institute of Technology, and the PhD degree in computing from Deakin University. He is currently the dean of the College of Computer and Information Science & College of Software, Southwest University, Chongqing, China, and a senior lecturer with Deakin University, Australia. He authored and co-authored more than 100 refereed papers in international journals or conference proceedings, six monographes or textbooks published by Springer. His research interests include multi-agent system, bio-inspired AI, and big data statistics and analysis.

Zhen Wang received the PhD degree from Hong Kong Baptist University, Hong Kong, in 2014. Since 2014, he has been a JSPS research fellow in the Interdisciplinary Graduate School of Engineering Sciences, Kyushu University, Fukuoka, Japan. He has authored/co-authored more than 80 research papers and four review papers with more than 4,000 citations. His current research interests include complex networks, evolutionary game, and data science.

Zhili Zhou received the MS and PhD degrees in computer application from the School of Information Science and Engineering, Hunan University, in 2010 and 2014, respectively. He is an assistant professor with the Nanjing University of Information Science and Technology. His current research interests include network science and data mining.