2013 IEEE Congress on Evolutionary Computation
June 20-23, Cancún, México

Community Detection Using Ant Colony Optimization

Chang Honghao, Feng Zuren
Systems Engineering Institute
State Key Laboratory for Manufacturing Systems
Xi'an Jiaotong University
Xi'an, China

Ren Zhigang
Autocontrol Institute
State Key Laboratory for Manufacturing Systems
Xi'an Jiaotong University
Xi'an, China

***Abstract*—Many complex networks have been shown to have community structure. How to detect the communities is of great importance for understanding the organization and function of networks. Due to its NP-hard property, this problem is difficult to solve. In this paper, we propose an Ant Colony Optimization (ACO) approach to address the community detection problem by maximizing the modularity measure. Our algorithm follows the scheme of max-min ant system, and has some new features to accommodate the characteristics of complex networks. First, the solutions take the form of a locus-based adjacency representation, in which the communities are coded as connected components of a graph. Second, the structural information is incorporated into ACO, and we propose a new kind of heuristic based on the correlation between vertices. Experimental results obtained from tests on the LFR benchmark and four real-life networks demonstrate that our algorithm can improve the modularity value, and also can successfully detect the community structure.**

I. Introduction

In recent decades, complex networks have received enormous amounts of attention from physicists, computer scientists, sociologists and mathematicians [1]–[3]. Complex networks theory has been applied successfully to topics of many kinds, such as the Internet [4], biology [5] and economy [6]. Many real-life networks are inhomogeneous, consisting of subgroup structures, within which the connections are dense, but between which the connections are sparse. This property is called *community structure* [7] in network science literature. How to detect and analyze the community structure in networks is of great significance for investigating the organization and function of complex systems.

Researchers have been working on the communities in networks for a long time. Historically, community detection in networks is closely related to the problems of graph partitioning in computer science and hierarchical clustering in sociology [8]. In recent years, a lot of effort has been made to develop new approaches and algorithms to detect and quantify community structure in complex networks. Various kinds of methods have been proposed, based on centrality measures [7], random walks [9], eigenvectors [10], spin models [11], and many other methods [12].

This paper focuses on the modularity optimization approach, which is by far the most used and the best known approach to detect the community structure in a network. One remarkable advantage of this method is that one does not need to have *a priori* knowledge about how many communities are there in the network. Since its introduction to the community detection problem [13], modularity has become an essential element in quite a lot of community detection algorithms. However, modularity optimization is not trivial. The number of ways to partition n vertices into k non-empty groups is given by the Sterling number of the second kind $S_n^{(k)}$, and hence the number of distinct community divisions is the Bell number $B_n = \sum_{k=0}^{n} S_n^{(k)}$, which grows at least exponentially in n [14]. Therefore, exhaustive optimization of the modularity is practically infeasible for networks larger than 70 vertices. Fortunately, several approximate methods have obtained pretty good results, including greedy optimization [15], genetic algorithms [16], multiobjective optimization [17], and so forth.

The proposed Ant Colony Optimization (ACO) [18] approach for community detection follows the framework of max-min ant system (MMAS) [19]. In this algorithm, each solution representing a community partition is recorded in the form of locus-based adjacency representation, and a new kind of heuristic is employed to construct these solutions. This heuristic is a similarity measure based on the structural equivalence of vertices, and is modified to accommodate the characteristics of the community detection problem. Compared with other algorithms, our algorithm not only reaches better modularities, but also can reveals communities close to the intrinsic network structure.

The rest of this paper is organized as follows: Section 2 is a review of the definition and modularity of the community structure. Then, in Section 3, we describe the new kind of heuristic information, the locus-based representation, and the MMAS algorithm adopted to find meaningful community structure. With experimental results on computer-generated networks and real-life networks, we demonstrate the effectiveness of our method in Section 4. The conclusions are drawn in Section 5.

II. Community structure and modularity

A network with n vertices (or nodes) can be modeled as a graph $G = (V, A)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is the set of vertices, and A is an $n \times n$ adjacency matrix representing connections (or edges) between vertices. The elements in A

978-1-4799-0454-9/13/$31.00 ©2013 IEEE

are real numbers. For an unweighted graph, the element A_{ij} of A is 1 if there is an edge connecting vertex v_i and vertex v_j, otherwise A_{ij} equals 0. The degree of a vertex v_i is denoted as k_i, and k_i equals to the number of edges incident to v_i. Community structure of the network can be represented in the form of a membership vector $C = \{c_1, c_2, \ldots, c_n\}$, where $c_i \in \{1, 2, \ldots, N\}$ is the index of the community to which vertex v_i belongs, with N the number of communities. We only consider the unweighted situation in this paper, however, applying our algorithm to weighted graphs is straightforward.

When tackling the community detection problem, one needs to figure out the definition of a community. Intuitively, the edge density should be high within a community, and low between communities, but this notion is ambiguous. No formal definition has been generally accepted in the network science community. This fact brings much difficulty to the community detection problem. Here, we introduce two kinds of definitions based on different scopes: local and global.

The local definitions focus on the single community and its direct neighbors. In [20], Radicci et al. proposed two local definitions of community, in a strong sense and a weak sense, respectively. The *strong community* is a subgraph such that each vertex has more connections within the community than with the rest of the graph. This definition can be relaxed into the so called *weak community*, where the sum of all degrees within the subgraph is larger than the sum of all degrees toward the rest of the network. According to this definition, a strong community is also a weak community, but the reverse is not true.

Opposite to the local definitions, the global one takes the network as a whole, and uses the *modularity* [13] as a criterion to identify the community structure. The concept of modularity is inspired by the idea that no community structure is expected to be found in a random graph. By comparing the fraction of edges within communities and the expected fraction of edges if the edges are drawn randomly between vertices without regard to the community structure, we can identify whether the network has community structure or not, and modularity is defined as the difference between the two quantities.

In an edge-randomized network with m edges, the probability for vertices v_i and v_j, with degrees k_i and k_j respectively, to be connected, can be derived rigorously. To form an edge, we need to pick a pair of *stubs* (i.e. half-edges) from the $2m$ stubs, so the probabilities of choosing a stub associated with vertex v_i is $p_i = k_i/2m$. The probability of a connection between v_i and v_j is given by $p_i p_j$, which equals to $k_i k_j / 4m^2$. Hence, the expected number of edges connecting vertices v_i and v_j is $a_{ij} = 2m p_i p_j = k_i k_j / 2m$. Then, the actual number of edges between v_i and v_j minus the expected number of edges is $A_{ij} - k_i k_j / 2m$, and the modularity Q is defined as the sum of this quantity over each pair of vertices that fall into the same community. We can write that

$$Q = \frac{1}{2m} \sum_{v_i, v_j \in V} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j) \qquad (1)$$

where $\delta(\cdot, \cdot)$ is the Kronecker delta function, $1/2m$ is only a normalizing constant, and c_i is the community index of v_i. If there are more number of edges within communities than the expected number in a randomized network then the modularity value Q would be greater than 0, and practically, modularities larger than 0.3 indicate significant community structure in networks.

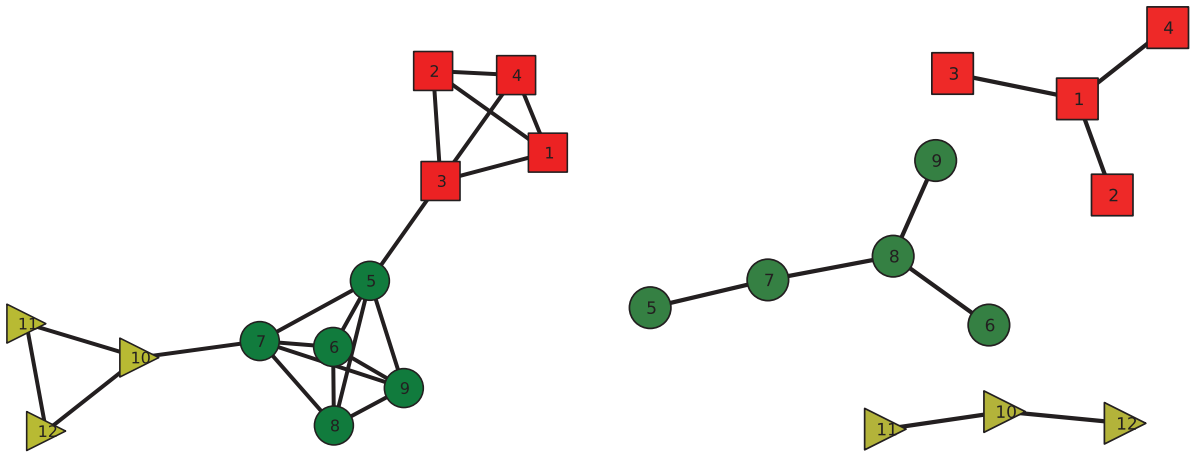
III. Algorithm

Ant Colony Optimization (ACO), a metaheuristic inspired by the foraging behavior of ant colonies [18], is a population-based approach for solving hard combinatorial problems, and has been successfully applied to various kinds of hard problems, like traveling salesman problem [21], quadratic assignment problem [19], and vehicle routing problem [22], to name a few. The ACO metaheuristic consists of a colony of artificial ants constructing solutions iteratively. The solution construction is guided by two kinds of information: one is the domain-based heuristic information and the other is the artificial pheromone trail. The ants communicate in an indirect way by means of the pheromone trail, which is modified to reflect the information accumulated during the search for a good solution. In 1996, the first ACO algorithm called Ant System (AS) was applied to solve the travelling salesman problem [23]. Since then, a lot of effort has been devoted to improving the performance of ACO algorithms. The Ant Colony System (ACS) [21], Max-Min Ant System (MMAS) [19], and the hyper-cube framework for ACO [24] are some of the variants that lead to better solutions than AS.

Before using the ACO to solve the community detection problem, one has to figure out a few things. What comes first is how to represent a solution for this problem. Next is how to construct a solution. Then, the initialization and update of the pheromone trail has to be determined. In this section, we will describe our algorithm from the aspects listed above.

A. Solution representation

As mentioned above in Section 2, a community is a subgraph of the network. Thus, detecting community structure in a network is equivalent to finding a group of subgraphs that reveals the best partition of the network. So the solution to the community detection problem can be formulated as a vector with n elements, representing a graph, whose connected component is equivalent to a community. Both indices and elements of the vector correspond to the vertices in the graph $G = (V, A)$. For example, if the ith element in the vector equals j, then this relation can be interpreted as a link between vertices v_i and v_j, meaning v_i and v_j are in the same connected component and, consequently, in the same community. The solution vector only describes the edges. To fully reveal the community structure, a decoding process is necessary and can be done through a breadth-first search (BFS) or a depth-first search (DFS) in linear time. After decoding, we get a membership vector representing the affiliation of each vertex. This kind of representation is called the locus-based adjacency representation, and has been employed by [17], [25] for multiobjective clustering. Figure 1a shows a network



(a) (b)

Index	1	2	3	4	5	6	7	8	9	10	11	12
Element	2	1	1	1	7	8	8	6	8	11	10	10

(c)

Index	1	2	3	4	5	6	7	8	9	10	11	12
Element	1	1	1	1	2	2	2	2	2	3	3	3

(d)

Fig. 1. (a) A hypothetical network with 3 communities. (b) The locus-based representation for a solution. (c) Three connected components representing the three communities. (d) Membership vector of the hypothetical network.

with 12 vertices partitioned into 3 groups, drawn as triangles, squares and circles, respectively. A possible solution vector to the community detection problem of the network in Figure 1a is shown in Figure 1c, and the corresponding subgraphs is depicted in Figure 1b. In Figure 1d, the membership vector is presented.

B. Solution construction

When applying ACO to the community detection problem, we formulate the solution as a set of n edges (some of which may be duplicated) for a network with n vertices, as mentioned in Section II. Hence the edges are used as solution components, and each edge (i, j) is associated with a pheromone trail τ_{ij} and heuristic information η_{ij}. The pheromone trail and heuristic information respectively indicate the accumulated and *a priori* preferences of choosing edge (i, j) as a solution component.

Generally, the solution construction starts with an empty solution, and then solution components are added iteratively on a probabilistic basis until the solution is completed. For the case of community detection, at construction step i, we choose an edge (i, j) from the edges incident to v_i according to the probability $p(i, j)$:

$$p(i,j) = \frac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum_{v_q \in \mathcal{N}_i}(\tau_{iq})^{\alpha}(\eta_{iq})^{\beta}} \qquad \text{if } v_j \in \mathcal{N}_i \tag{2}$$

where $\mathcal{N}_i$ denotes the neighborhood of vertex v_i, and parameters α and β determine the relative importance of the pheromone trail and the heuristic information. After n construction steps, a complete solution containing n edges is generated. The modularity of this community partition calculated from the solution is essential to determine how the pheromone trails are updated.

C. Heuristic information

It is generally accepted that using heuristic information to direct the probabilistic solution construction can play an important role in ACO algorithms [18]. The heuristic information provides an approach to exploit the prior knowledge about one particular problem, and we try to incorporate this knowledge into our community detection algorithm.

As the old saying goes, "birds of a feather flock together," intuitively, the vertices in the same community are similar to each other, so the similarity measure between vertices can be used as heuristic information. There are several kinds of similarity measures to describe the relations between vertices in an Euclidean space [26], such as the Euclidean distance, the Manhattan distance, and the L_∞-norm. But, usually, a generic graph cannot be embedded in an Euclidean space, and the similarity must be inferred from the structural equivalence [12]. In this paper, we use the Pearson correlation between vertices as the similarity measure. The Pearson correlation $C(i, j)$ between vertices v_i and v_j is defined as follows:

$$C(i,j) = \frac{\sum_{v_l \in V}(A_{il} - \mu_i)(A_{jl} - \mu_j)}{n\sigma_i\sigma_j} \tag{3}$$

where A_{il} is the lth element of the ith row in the adjacency matrix, $\mu_i = \sum_l A_{il}/n$ is the average, and $\sigma_i = \sqrt{\sum_l (A_{il} - \mu_i)^2/n}$ is the standard deviation.

If vertices v_i and v_j are very similar in a structural perspective, then $C(i, j)$ would be close to 1. Otherwise, $C(i, j)$ would be close to -1. Since the Pearson correlation may take negative value, it cannot be directly used in ACO. We apply the logistic function to the Pearson correlations defined above to make the outputs positive, and the outputs are incorporated

into the ACO algorithm as heuristic information:

$$\eta_{ij} = \frac{1}{1 + e^{-C(i,j)}} \tag{4}$$

The more similar vertices v_i and v_j are, the larger value η_{ij} would take, and the more probably edge (i, j) is to be taken as a solution component.

D. Pheromone trail management

Our method follows the pheromone trail update mechanism used in MMAS [19]. All the pheromone trails are initialized to an arbitrary high value at the beginning of the algorithm, in order to favor the exploration of solutions during the early part of the searching process.

In each iteration, after all ants have completed their solution construction, the pheromone trails are updated in two opposite ways. Firstly, all the pheromone trails are reduced by a constant factor, and this procedure helps ants to "forget" the bad solutions and favors the exploration of new solutions. Then, a reinforcement procedure is performed based on the solution quality. During the reinforcement stage, only one ant is used to deposit pheromone on the edges of the network. In this paper, we use the *iteration-best* solution to add pheromone. The iteration-best solution is the solution with the largest value of modularity in the current iteration, and is denoted as s^{ib}. Only edges in the iteration-best solution receive pheromone deposition. So the pheromone trail update rule can be formulated as follows:

$$\tau_{ij} \leftarrow \rho\tau_{ij} + \Delta\tau_{ij} \tag{5}$$

where the parameter ρ $(0 \leq \rho < 1)$ is the pheromone persistence, $\Delta\tau_{ij}$ is the amount of pheromone put on the edge (i, j), and it can be represented as:

$$\Delta\tau_{ij} = \begin{cases} \mathcal{M}(s^{ib}) & \text{if edge } (i,j) \text{ is in } s^{ib}, \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

where $\mathcal{M}(s^{ib})$ is the modularity of the solution s^{ib}. The greater the modularity is, the more amount of pheromone is put on the corresponding edges.

Aiming to avoid stagnation, we limit the range of possible pheromone amount on each edge to an interval $[\tau_{\min}, \tau_{\max}]$ as suggested by MMAS. The upper limit of pheromone trail is set to an estimate of the asymptotically maximum value, i.e., $\tau_{\max} = \mathcal{M}(s^{gb})/(1-\rho)$, where s^{gb} is the solution that has the largest value of modularity found so far since the beginning of the trial. The lower limit $\tau_{\min}$ is set to $\tau_{\min} = \varepsilon\tau_{\max}$, where ε $(0 < \varepsilon < 1)$ is a rate coefficient. Once a new s^{gb} is found, the limits $\tau_{\max}$ and $\tau_{\min}$ are updated accordingly. And the complete ACO algorithm for community detection is shown in Figure 2.

Input: A network modeled by the graph $G = (V, A)$
Output: A community partition that maximize the modularity

```
Set parameters and initialize pheromone trails
while (termination condition not met) do
    S ← ∅    // S represents the whole ant colony
    repeat
        Construct a new solution s according to Eq. 2
        S ← S ∪ s
    until |S| = n_a
    Update the iteration-best (s^ib) and the global-best (s^gb)
    Compute pheromone trail limits τ_max and τ_min
    Update pheromone trails
end while
```

Fig. 2. ACO algorithm for community detection

IV. Applications

To study the effectiveness of our algorithm, we performed tests on a series of synthetic benchmarks and four real-life networks and compared with six state-of-the-art algorithms, five from the network science literature, one from the evolutionary computation literature. The five algorithms from network science are as follows: the first is proposed by Newman (denoted as eigenvector),using eigenvectors of a modularity matrix to maximize the modularity [10], the second is Clauset et al.'s fast greedy modularity optimization algorithm (denoted as fastgreedy) [15], the third is proposed by Blondel et al. (denoted as multilevel), following a hierarchical approach to optimize the modularity [27], the fourth is Pons and Latapy's algorithm based on random walking [9], denoted as randomwalk, and the fifth is an algorithm trying to find communities via a spin glass model (denoted as spinglass) [11]. The one method from evolutionary literature is MOGA-Net, which is a multiobjective algorithm, developed by Pizutti [17].

We implemented all algorithms in MATLAB, except for MOGA-Net, and the results for MOGA-Net come from [17]. The experiments were performed on a PC with a 2.0 GHz Pentium 4 CPU and 2 GB RAM under Windows 7 OS. It is known that parameters play a significant role in ACO [18], and inappropriate parameter-setting may lead to undesirable performance. However, finding a good parameter configuration can be time consuming, and in this paper our aim is to show that our algorithm is effective. So we only set the parameters to typical values. Specifically, we set $\alpha = 1$, $\beta = 2$, $\rho = 0.8$, $\varepsilon = 0.005$, and the number of ants $n_a = 30$. We use maximum number of iterations as termination condition, and unless otherwise mentioned, the maximum number of iterations is set to 100.

A. Evaluation Criteria

Since many algorithms are modularity-optimization-oriented approaches, the modularity is definitely a necessary criterion for the performance evaluation. However, the modularity measure can not reflect the difference or similarity between the found community structure and the real structure. To overcome this disadvantage, we adopted a measure called *normalized mutual information* (NMI) from information theory [28], which has been proved to be more appropriate and reliable.

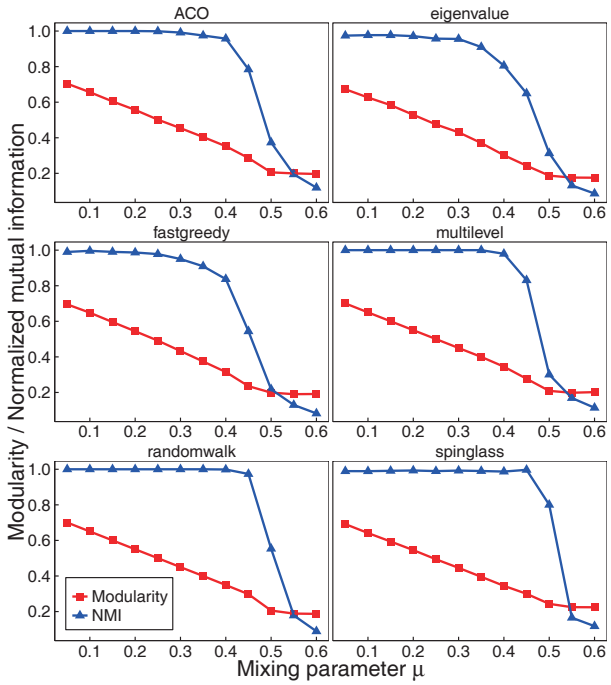


Fig. 3. Tests on the LFR benchmark.

Let $X = \{x_1, \ldots, x_g\}$ and $Y = \{y_1, \ldots, y_h\}$be two sets of communities. The entropy of X can be expressed as $H(X) = -\sum_{i=1}^{g} \frac{n_i^x}{n} log(\frac{n_i^x}{n})$, where n_i^x represents the number of vertices in community x_i. The agreement between partitions X and Y is measured by the *mutual information*, defined as

$$I(X,Y) = \sum_{i=1}^{g}\sum_{j=1}^{h} \frac{n_{ij}^{xy}}{n} log\left(\frac{\frac{n_{ij}^{xy}}{n}}{\frac{n_i^x}{n}\cdot\frac{n_j^y}{n}}\right). \quad (7)$$

where n_{ij}^{xy} denotes the number of shared vertices between communities x_i and y_j. The normalized mutual information between two partitions X and Y can be formulated as

$$NMI(X,Y) = \frac{2I(X,Y)}{H(X)+H(Y)}. \quad (8)$$

If the partitions are identical, the NMI equals 1, whereas it takes a value of 0 if the partitions are independent.

B. Synthetic networks

The so called *planted ℓ-partition model* [29] has been adopted in many occasions. The model partitions a network with $n = g \cdot \ell$ vertices into ℓ communities, each with g vertices. For each vertex, the probability of a link between itself and other vertices in the same community is p_{in}, and the probability of being linked to a vertex in other communities is p_{out}. The network keeps a community structure as long as $p_{in} > p_{out}$. One of the most popular version of planted ℓ-partition model is the LFR benchmark proposed by Lancichinetti et al. [30]. The LFR benchmark consists of 128 vertices with an average degree of 16, and the vertices are divided into four communities with 32 vertices each. Every vertex shares a fraction μ of its edges with vertices in other communities, and $1-\mu$ with vertices in its own community, and $0 \leq \mu \leq 1$ is called the mixing parameter. When $\mu < 0.5$, there are more edges within communities than between communities.

Using the modularity and the normalized mutual information between true partitions and detected ones as evaluation metrics, we compared our method with the five community detection algorithms coming from network science literature. Figure 3 shows the results obtained from tests on the LFR benchmark, each point of every curve corresponding to an average of the 20 implementations. The modularities of detected community partitions obtained from all six algorithms are almost identical. And the trends are generally the same: for small mixing parameters, the community structure is quite obvious, and both the modularity and NMI measures are pretty good. As μ increases, the community structure become more and more vague, and both measures decrease substantially. The results obtained by our method is slightly worse than the randomwalk and spinglass algorithms, almost as good as the multilevel method, and better than the eigenvector and fastgreedy algorithms. Even when $\mu = 0.4$, the normalized mutual information is still greater than 0.8.

C. Real-life networks

We also applied our algorithm to four well known real-life networks, including the Zachary's karate club, the bottlenose dolphins social network, the American College Football, and the Krebs' books on US politics. We compared the results with algorithms coming from both the network science and evolutionary computation literature.

The Zachary's karate club network was recorded by Zachary during a two-year study of the social interactions between 34 members of a karate club in an American university [31]. The club was divided into two groups in the end as a result of an internal dispute between the club's administrator and its principle karate teacher. This network has become a benchmark in recent years to test the accuracy of community detection algorithms. In Figure 4, the karate club network is shown, the shapes of the vertices indicate the original division, and the colors represent the partition found by our algorithm. Our algorithm divided each of the original two communities into two subgroups, and got a modularity $Q = 0.420$, which is also the optimal value of modularity on this network.

The second network is the Bottlenose Dolphins social network compiled by Lusseau et al. [32] from seven-year observation of a community of 62 dolphins off Doubtful Sound, New Zealand. The vertices represent the dolphins, and a tie between a pair of dolphins means that the association between this pair occurred more often than expected. The temporary disappearance of one dolphin made the dolphin community break into two parts. The modularity obtained by our method is 0.529, also a global optimum.

In the American College Football network (here, football means American football, not soccer), vertices are Division I college football teams, and two teams are connected if

TABLE I
RESULTS ON ZACHARY'S KARATE CLUB AND BOTTLENOSE DOLPHINS

Algorithm	Zachary's Karate Club		Bottlenose Dolphins	
	Modularity	NMI	Modularity	NMI
ACO	**0.420**	0.687	**0.529**	0.587
MOGA-Net	0.416	0.602	0.505	0.506
eigenvector	0.393	0.677	0.491	0.449
fastgreedy	0.281	**0.693**	0.495	0.573
multilevel	0.419	0.587	0.519	0.511
spinglass	0.419	0.587	0.525	**0.590**
randomwalk	0.353	0.504	0.489	0.537

TABLE II
RESULTS ON AMERICAN COLLEGE FOOTBALL AND BOOKS ABOUT US POLITICS

Algorithm	American College Football		Books about US politics	
	Modularity	NMI	Modularity	NMI
ACO	**0.605**	0.890	**0.527**	**0.560**
MOGA-Net	0.515	0.775	0.518	0.537
eigenvector	0.488	0.685	0.467	0.520
fastgreedy	0.577	0.762	0.502	0.531
multilevel	0.602	0.855	0.520	0.512
spinglass	0.601	**0.904**	0.526	0.495
randomwalk	0.604	0.887	0.520	0.543

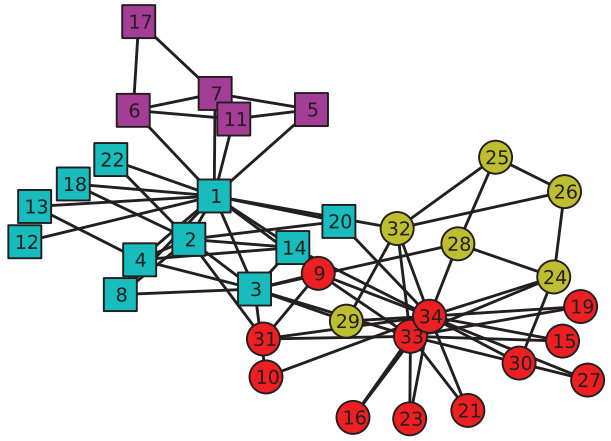


Fig. 4. Zachary's karate club network.

they play each other in the regular season [7]. The 115 teams are divided into 12 conferences, and generally more intraconference games are scheduled than the interconference games. So the community structure should be obvious. And our algorithm got a partition with a modularity of 0.605 and an NMI of 0.890.

The last example is a network of books about US politics, compiled by V. Krebs [33]. The vertices represent 105 books at the time of the 2004 presidential election from Amazon.com, and edges between books represent frequent copurchasing of books by the same buyer. Based on their political alignment (liberal or conservative), the books were divided by Newman into two categories, except for a small number of books that were bipartisan or centrist, or had no clear affiliation. The modularity obtained by our algorithm is 0.527.

Both values of modularity and normalized mutual information are shown in Tables I and II, and for every instance, the best modularity and NMI are presented in boldface. Tables I and II clearly show that our method performed very well on the real-life networks. In all four cases, our algorithm gives the best modularity measures. For the books network, our algorithm also gives the best NMI result, and in the other three networks, our algorithm gets the second best NMI results, only outperformed by a slim margin. For any of these four examples, no other algorithms have better results in both measures at the same time.

V. CONCLUSIONS

This paper has presented an ant colony optimization approach for the community detection problem. Using the locus-based adjacency presentation, and with the help of a new kind of heuristic information, we adopt the MMAS framework to detect community structure in networks by modularity optimization. The results from experiments on both synthetic and real-life networks is encouraging: our algorithm leads to better modularities than existing algorithms, and the normalized mutual information results are better than most of those obtained from the existing ones. This study also shows that ACO has great potential in solving community detection problems. We only deal with static networks in this paper, however, many networks, especially social networks, are changing temporally. Therefore, developing algorithms that can dynamically detect community structure will be our focus in the future.

ACKNOWLEDGMENTS

This work is partially supported by the National Natural Science Foundation of China under Grant No. 61105126 and the Doctoral Foundation of Education Ministry of China under Grant No. 20100201110031.

REFERENCES

[1] R. Albert and A.-L. Barabási, "Statistical mechanics of complex networks," *Rev. Mod. Phys.*, vol. 74, pp. 47–97, 2002.

[2] M. E. J. Newman, "The structure and function of complex networks," *SIAM Review*, vol. 45, no. 2, pp. 167–256, 2003.

[3] S. H. Strogatz, "Exploring complex networks," *Nature*, vol. 410, no. 6825, pp. 268–276, 2001.

[4] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law relationships of the internet topology," *Comput. Commun. Rev.*, vol. 29, no. 4, pp. 251–262, 1999.

[5] R. Guimerà and L. A. Nunes Amaral, "Functional cartography of complex metabolic networks," *Nature*, vol. 433, no. 7028, pp. 895–900, 2005.

[6] C. A. Hidalgo, B. Klinger, A.-L. Barabási, and R. Hausmann, "The product space conditions the development of nations," *Science*, vol. 317, no. 5837, pp. 482–487, 2007.

[7] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proc. Natl. Acad. Sci.*, vol. 99, no. 12, pp. 7821–7826, 2002.

[8] M. E. J. Newman, "Communities, modules and large-scale structure in networks," *Nature Physics*, vol. 8, no. 1, pp. 25–31, 2012.

[9] P. Pons and M. Latapy, "Computing communities in large networks using random walks," *ArXiv Physics e-prints*, 2005.

[10] M. E. J. Newman, "Finding community structure in networks using the eigenvectors of matrices," *Phys. Rev. E*, vol. 74, no. 3, 2006.

[11] J. Reichardt and S. Bornholdt, "Statistical mechanics of community detection," *Phys. Rev. E*, vol. 74, no. 1, p. 016110, 2006.

[12] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3, pp. 75–174, 2010.

[13] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Phys. Rev. E*, vol. 69, no. 2, p. 026113, 2004.

[14] R. L. Graham, D. E. Knuth, and O. Patashnik, *Concrete mathematics*. Reading, MA: Addison-Wesley Publishing Company, 1989.

[15] A. Clauset, M. E. J. Newman, and C. Moore, "Finding community structure in very large networks," *Phys. Rev. E*, vol. 70, no. 6, p. 066111, 2004.

[16] S. Li, Y. Chen, H. Du, and M. W. Feldman, "A genetic algorithm with local search strategy for improved detection of community structure," *Complexity*, vol. 15, no. 4, pp. 53–60, 2010.

[17] C. Pizzuti, "A multiobjective genetic algorithm to find communities in complex networks," *Evolutionary Computation, IEEE Transactions on*, vol. 16, no. 3, pp. 418–430, 2012.

[18] M. Dorigo and T. Stützle, *Ant Colony Optimization*. Cambridge, MA: MIT Press, 2004.

[19] T. Stüzle and H. H. Hoos, "Max-min ant system," *Future Generation Computer Systems*, vol. 16, no. 8, pp. 889–914, 2000.

[20] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi, "Defining and identifying communities in networks," *Proc. Natl. Acad. Sci.*, vol. 101, no. 9, pp. 2658–2663, 2004.

[21] M. Dorigo and L. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *Evolutionary Computation, IEEE Transactions on*, vol. 1, no. 1, pp. 53–66, 1997.

[22] B. Bullnheimer, R. Hartl, and C. Strauss, "An improved ant system algorithm for the vehicle routing problem," *Ann. Oper. Res.*, vol. 89, pp. 319–328, 1999.

[23] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 26, no. 1, pp. 29–41, 1996.

[24] C. Blum and M. Dorigo, "The hyper-cube framework for ant colony optimization," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 34, no. 2, pp. 1161–1172, 2004.

[25] J. Handl and J. Knowles, "An evolutionary approach to multiobjective clustering," *Evolutionary Computation, IEEE Transactions on*, vol. 11, no. 1, pp. 56–76, 2007.

[26] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification (2nd Edition)*. New York: Wiley-Interscience, 2000.

[27] V. Blondel, J. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, p. P10008, 2008.

[28] L. Ana and A. Jain, "Robust data clustering," in *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, vol. 2, 2003, pp. II–128 – II–133 vol.2.

[29] A. Condon and R. Karp, "Algorithms for graph partitioning on the planted partition model," *Random Structures and Algorithms*, vol. 18, no. 2, pp. 116–140, 2001.

[30] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Phys. Rev. E*, vol. 78, no. 4, p. 046110, 2008.

[31] W. Zachary, "An information flow modelfor conflict and fission in small groups1," *Journal of Anthropological Research*, vol. 33, no. 4, pp. 452–473, 1977.

[32] D. Lusseau, K. Schneider, O. Boisseau, P. Haase, E. Slooten, and S. Dawson, "The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations," *Behavioral Ecology and Sociobiology*, vol. 54, no. 4, pp. 396–405, 2003.

[33] M. E. J. Newman, "Modularity and community structure in networks," *Proc. Natl. Acad. Sci.*, vol. 103, no. 23, pp. 8577–8582, 2006.