2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

Ant Colony Clustering Based on Sampling for Community Detection

Xiangjing Song, Junzhong Ji, Cuicui Yang
College of Computer Science
Beijing University of Technology
Beijing, China
Email: jjz01@bjut.edu.cn

Xiuzhen Zhang
School of Computer Science and IT
RMIT University
Melbourne, Australia
Email: xiuzhen.zhang@rmit.edu.au

***Abstract*—Community structure detection in large-scale complex networks has been intensively investigated in recent years. In this paper, we propose a new framework which employs the ant colony clustering algorithm based on sampling to discover communities in large-scale complex networks. The algorithm firstly samples a small number of representative nodes from the large-scale network; secondly it uses the ant colony clustering algorithm to cluster the sampled nodes; thirdly it assigns the un-sampled nodes into the detected communities according to the similarity metric; finally it merges the initial clustering result to sustainably increase the modularity function value of the detection results. A significant advantage of our algorithm is that the sampling method greatly reduces the scale of the problem. Experimental results on computer-generated and real-world networks show the efficiency of our method.**

I. Introduction

Many systems in real world exist in the form of networks that contain complex interactions between individuals, such as biological networks, social networks, Web networks, etc, which are known as complex networks. A complex network is typically represented as an undirected graph G(V, E), where V is the set of nodes and E is the set of edges. And a distinguishing property of the complex networks is community structure, which means that nodes within a group are much more connected to each other than to the rest of the network [1].

A lot of methods have been applied to complex networks to detect community structure, and there are also some special researches on community detection in large-scale networks proposed along with the expansion of complex networks. For instance, Newman proposed a fast algorithm for detecting community structure based on Q metric, which is a classic algorithm named Fast Newman (FN) [1]. He et al. presented a new ant colony optimization for community detection in large networks, named MACO, the approach uses ants to propagate the label of its current position to others according to a simulated annealing idea, whose purpose is to locally optimize modularity Q, and introduces the thought of "layer and rule" to improve the performance [2]. Jin et al. gave a genetic algorithm with local search (named LGA) for community detection, and the idea of local search based on the analysis on local monotonicity of function Q, meanwhile, to produce the accurate and diverse initial population, the algorithm adopted a label propagation based method [3]. Raghavan et al. investigated a simple label propagation algorithm, known as LPA, which has near linear run-time. And in this algorithm every node is initialized with a unique label, and each node adopts the label that most of its neighbors currently have at every step, this process is repeated till all nodes have a label that the maximum of their neighbors have [4]. The research referred above employs different search mechanisms to deal with the problem of community detection in large-scale complex networks effectively, but these methods are not sufficient to completely solve the problem along with the expansion of network scale.

The community detection in large-scale complex networks is different from the clustering problem of large-scale datasets, and the main difference is they aim at different datasets which have different topological properties. However, they are both essentially the clustering problems, and both face with the contradiction that the scale of data sets expands gradually in realistic society, but the speed of clustering method is slower. For clustering problem of large-scale datasets, there are some useful researches; for instance, Zhou et al. developed a fast DBSCAN algorithm, named FDBSCAN, which uses only a small number of representative points as seeds to expand the cluster [5]. Guha et al. raised CURE (Clustering Using Representatives) algorithm, and the algorithm accomplished clustering by combining the random sampling and partitioning [6]. However, as far as we know, there is no research of using the similar sampling method to detect the communities from large-scale complex networks till date. Hence, we employ the sampling idea, and present a new Ant Colony Clustering algorithm based on Sampling for community detection in large-scale complex networks, called ACCS, and experimental results show that the algorithm is fast and can gain good clustering results.

The rest of this paper is organized as follows. Section II presents the main idea, steps and description of ACCS algorithm. Section III presents and analyzes the experimental results. And at last, section IV concludes this paper.

II. Algorithm

A. The main idea

In [7], we have proposed an Ant Colony Clustering algorithm based on Fitness perception and Pheromone diffusion (called as ACC-FP) to detect community in complex networks. Although the time complexity of ACC-FP is better than typical algorithms, it is still hard to satisfy the needs of community

978-1-4799-1488-3/14/$31.00 ©2014 IEEE

detection in large-scale complex networks. To further improve the time performance, we propose the ACCS algorithm in this paper, and the basic thought is as follows.

At the beginning of ACCS algorithm, it selects some representative nodes from the initial network as a new network according to the degree, so as to reduce the scale of the problem. Secondly it uses the ant colony clustering algorithm to detect communities for the new network. And in this phase, each sampled node is initialized to a community and randomly distributed as an ant on the grid. Thereafter, the community detection result will be evolved in a number of cycles. In each cycle, each of the ants makes use of the perception of the environment to move to a new location or stay in the original location. Ant colony moving at each cycle forms a new community result, whose quality is evaluated and employed to update the pheromone of nodes. The evolution process is repeated till all ants find and keep the most comfortable positions, and the community structure detection of the new network is accomplished. Then based on the community detection results obtained by above steps, this algorithm completes the assignment of remaining nodes according to the similarity between the un-sampled nodes and the formed community. Finally, we merge communities by sustainably increasing the Q value.

B. Sampling from the complex network

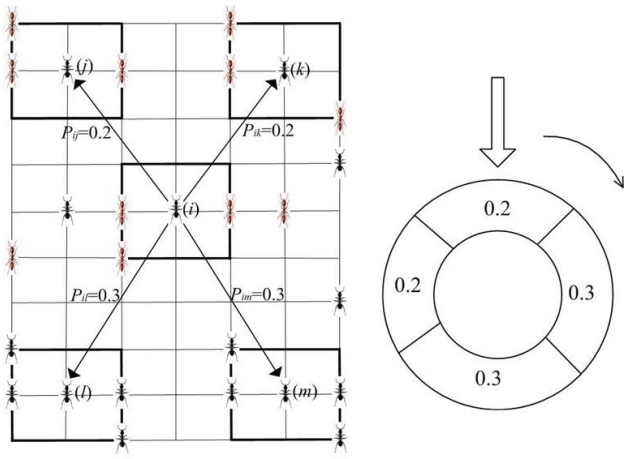
To reduce the scale of the problem, we use the sampling method to select some representative nodes which can depict the initial large-scale network. And the key question is whether we can find the similar community structure between the new network and the initial network. Ideally, we hope the sampled nodes contain some nodes of each community, and the nodes that belonging to different communities in the initial network can also in different communities in the new network. There are a lot of sampling methods in statistics that have been used in the clustering of large-scale datasets. Considering the topology characteristics of complex networks, the bigger value of the node degree is, the greater attraction the node has on the others [8]. Hence, we adopt a sampling method by means of the node degree. And the sampling rate R is defined as:

$$R = n/N \tag{1}$$

where N is the total nodes in the whole network, and n is the number of the sampled nodes. We order all nodes in reverse according to the node degrees, and choose the sampled nodes by the sampling rate.

C. Clustering the sampled nodes

1) Ant colony clustering algorithm: The basic principle of the ant colony clustering algorithm is as follows [7]. Each ant is a simple agent who represents a node of the sampled network. All ants have two states on a two-dimensional grid: the active state and the sleeping state. When the ant's fitness is low, it has a higher probability to wake up and stays in active state. It will leave its original position to search for a more comfortable position to sleep. When an ant locates in a comfortable position, it has a higher probability to sleep until the surrounding environment changes and activates the ant again. This process is repeated till the community structure is obtained. During the process of ant colony clustering, we proposed an ant moving strategy, into which we integrate the label propagation in ACCS.



a) The probability model. b) The roulette selection

Fig. 1. The sketch map of the ant moving strategy

Fig.1 shows the ant moving strategy, Fig.1 (a) gives an example to explain the probability model, and Fig.1 (b) presents the roulette selection. In Fig.1 (a), first we suppose that there are 11 red ants and 15 black ants on the grid, the ants with the same color denote that they are in the same community. The current position of ant i is uncomfortable, and it has edges link to the ants j, k, l, m. And the areas of figure shown in bold are the neighborhoods of the ant i, j, k, l, m in the grid. As ant i feels uncomfortable at the current position, it will turn into the active state, we select one node from all which link to node i as the objective node. In the process of choosing, we not only consider the combination of heuristic information that includes the common neighbor information and the degree of the node in the complex network, but also employ the aggregation pheromone to reflect the clustering situation in the grid; from this we can compute and get the probabilities in light of Eq.(2) to Eq.(4), and the value 0.2, 0.2, 0.3, 0.3 in Fig.1 (a) means that ant j, k, l, m all has the possibility to be the objective one. To determine the objective node, we use the roulette selection as shown in Fig.1 (b), and move ant i to an empty position in the neighborhood of the objective ant on the grid. In the end, let ant i be sleeping state.

$$p_{ij}(t) = \begin{cases} \frac{[\ _j(t)]\ [\ _j]}{\sum_{l\ Neighbor(i)} [\ _l(t)]\ [\ _l]}, & j \in Neighbor(i) \\ 0, & otherwise \end{cases} \tag{2}$$

$$\tau_j(t) = \sum_{k \in A} \frac{\tau_{k \to j}(t)}{d_{kj}} \tag{3}$$

$$\eta_j = \begin{cases} |c(i,j)|, & \sum_{l \in Neighbor(i)} |c(i,l)| \neq 0 \\ d(j), & otherwise \end{cases} \tag{4}$$

where $p_{ij}(t)$ represents the possibility that the ant i moves to an empty location of the ant j's neighborhood where there is an edge between two represented nodes in G(V, E), $\tau_j(t)$ is the quantity of aggregation pheromone laying on the node j at

time t(t is the number of iterations), A represents the node j's neighborhood in the grid, $\tau_{kj}(t)$ is the aggregation pheromone on the node j imposed by the node k's pheromone, and d_{kj} denotes the distance between two node j and k in the complex network; η_j represents a local heuristic information, $|c(i,j)|$ is the number of common neighbor nodes for the two linked nodes i and j, $d(j)$ is the degree of the node j; $Neighbor(i)$ is the neighbor node set of node i in the complex network, and parameters α and determine the relative importance of pheromone trail versus heuristic factor for the node j.

There are some other strategies which have been proposed in the clustering phase, such as the aggregation pheromone diffusion and updating model, the adaptive adjusting of fitness threshold value, please see the specific formulas and introductions in [7].

2) Label propagation of ants: At the beginning of ant colony clustering algorithm, each ant is seen as a community, and carries a different label denoting the community to which it belongs. In the process of evolution, after moving to the new location, each ant updates its label according to the neighbors which connect with it in the network and also in its neighborhood on the grid, then selects the label that is the maximum number of its neighbors belongs to.

The detail of ants' label propagation is in the following. Suppose the label of ant i at iteration t is $l_i(t)$, there are k_1 ants in ant i's neighborhood on the grid, and k_2 ants out of k_1 are connected with ant i. Then ant i at iteration t updates its label based on the labels of its neighbors that have already been updated in the current iteration and that are not yet updated in the last iteration. The formula of label updating is as follows:

$$l_i(t) = f(l_1(t), \cdots, l_j(t), l_{j+1}(t) \ldots, l_{k2}(t-1)) \quad (5)$$

where the function f on behalf of calculating the label that appears most.

3) Assigning the un-sampled Nodes: When the phase that clusters the sampled nodes is over, the next step is to assign the un-sampled nodes into the detected communities naturally. We use the similarity metric to evaluate the distance between nodes and communities, and find the most comfortable community that the node joins in, and the formula of the similarity is as follows:

$$S_i(J) = \sum_{k \in J} a_{ki} \quad (6)$$

where $S_i(J)$ denotes the similarity between the node i and the community J. a_{ki} represents the connectivity relationship between the node k that belongs to community J and the node i, and a_{ki} is equal to 1 when the edge e_{ki} exists, and zero otherwise. The formula counts all edges that the nodes in community J link with the node i. The larger the value of $S_i(J)$, the more similar the node with the corresponding community. Thus we select the largest value of $S_i(J)$ as the best community that the node i joins in, and assign the label of the community J to the node i.

4) Merging the communities: In general, after the un-sampled nodes assigning, we can consider the process of community detection in large-scale complex network is complete. However, it should be noted that if two communities connected by several nodes, which we don't choose in the phase of sampling, it is likely to lead to community split. So when accomplishing the assigning phase, we merge the communities with the modularity Q as the objective function. And the module function [1] is defined as:

$$Q = \sum_{s=1}^{M} \left[\frac{l_s}{L} - \left(\frac{d_s}{2L} \right)^2 \right] \quad (7)$$

where M is the number of communities, L is the number of edges in the network, l_s denotes the number of edges between nodes in community s, and d_s is sum of the degrees of nodes in community s. If the edges within-community in the detected result is less than the random one, the value of Q is negative. On the contrary, when the value of Q is close to 1, it means the corresponding community structure is very well. Thus, when we merging two communities, if Q which denotes the change of Q is greater than zero, then we merge the two communities; Otherwise, we will do nothing.

Algorithm 1 The ACCS algorithm

Input: Graph $G(V,E)$: a complex network
Output: C: the set of communities
1. Initialization
  Set parameters;
  Compute the degree of nodes in $G(V,E)$.
2. Sampling from the Complex Network
  For i = 1 to N
    Select n nodes from all as the new network according to the sampling method.
3. Clustering the Sampled Nodes
  For $t = 0$ to T $*T$ is the maximum number of iterations
    For $i = 1$ to n
      Set ant's label be i, and compute the value of fitness function;
      If ant i feels uncomfortable then
        Let ant i move to a new position according to the moving strategy;
        Update the label of the ant i.
4. Assigning the Un-sampled Nodes
  For $i = 1$ to $N - n$
    For $j = 1$ to M
    $*M$ denotes the number of detected communities $*$
      Compute the similarity between the un-sampled nodes and the communities;
      Record the maximum of the similarity and corresponding label;
      Update the label of the un-sampled node.
5. Merging the Communities
  For $i = 1$ to M
    For $j = 1$ to M
      Compute Q;
      If ($Q > 0$) then Merge the communities.
6. Output
  Return the communities for the large-scale complex network.

5) Algorithm description and complexity analysis: Summing up the above ideas, ACCS can be simply described as the following four steps: Sampling nodes from the complex network, clustering the sampled nodes, assigning the un-sampled nodes, and merging communities. The algorithm is

summarized in Algorithm 1. The time complexity of the sampling phase is $O(N + R \cdot N \cdot logN)$; the second phase is $O(T \cdot k' \cdot R \cdot N)(k' \ll R \cdot N)$, and you can see the specific analysis in [7]; the third phase is $O((N - R \cdot N) \cdot M)$; the last phase is $O(M \cdot logM)$, and $M \ll N$, so the $O(M \cdot logM)$ can be negligible. In conclusion, the complexity of the algorithm is $O(N+R \cdot N \cdot logN+T \cdot k' \cdot R \cdot N+(N-R \cdot N) \cdot M)$, and it can be simplified as $O(N(1+R \cdot logN+T \cdot k' \cdot R+M-R \cdot M))$. As $(1+T \cdot k' \cdot R+M-R \cdot M)$ is a constant, the complexity converges to $O(N \cdot R \cdot logN)$with the increase of N.

III. Experiment

In the experiment, we both use the computer-generated and real-world networks to perform our empirical study. First, we test the performance of ACCS on many computer-generated networks to illustrate the capability of the approach. Then, our algorithm is compared with FN [1] on seven large-scale real-world networks with unknown community structures.

In ACCS algorithm, there is only a new parameter which is the sampling rate R, the value of R is directly related to the number of sampled nodes, and thus has a great influence on the detection results. So in this paper, we set $R = 0.2$ in the computer-generated networks, and set different values of R in the real-world networks. The determination of the rest parameters can be seen in [7].

A. Computer-generated Networks

We adopt some random networks with known community structure, which have been used as benchmark datasets for testing complex network clustering algorithms [9]. This kind of random networks is defined as $LFR(N, k, \ , \phi, \varphi)$, where N is the number of nodes in a network, k is the average degree of nodes, is the exponent of the degree distribution, ϕ is the exponent of the community size distribution, and φ is a mixing parameter which is used to control the ratio of edges among different communities. When $\varphi > 0.5$, the neighbors of a node inside its community are less than the neighbors belonging to the rest groups, in this case we consider the random network don't have the community structure.

Moreover, here we employ two widely used accuracy measures, which are Fraction of Vertices Classified Correctly (FVCC) [1] and Normalized Mutual Information (NMI) [10]. The FVCC is a simple measure to evaluate the clustering accuracy while the NMI is adopted to estimate the similarity between the true partitions and the detected ones. For NMI, Given two partitions A and B of a network in communities, let C be the confusion matrix whose element C_{ij} is the number of nodes of community i of the partition A that are also in the community j of the partition B. The $NMI(A, B)$ is defined as follows:

$$NMI(A,B) = \frac{-2\sum_{i=1}^{C_A}\sum_{j=1}^{C_B} C_{ij} log(C_{ij}N/C_i\, C_j)}{\sum_{i=1}^{C_A}(C_i\, log(C_i/N)) + \sum_{j=1}^{C_B}(C_j log(C_j/N))} \quad (8)$$

$C_A(C_B)$ is the number of groups in the partition $A(B)$, C_i (C_j) is the sum of the elements of C in row i (column j),

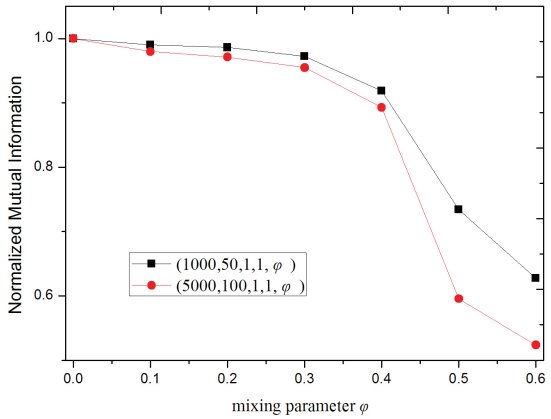


Fig. 2. The NMI performances

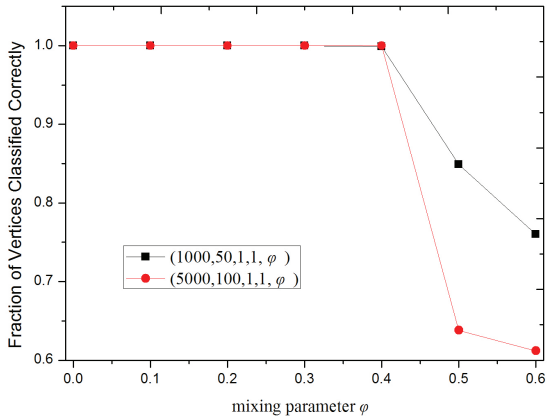


Fig. 3. The FVCC performances

and N is the number of nodes. If $A = B, NMI(A, B) = 1$. If A and B are completely different, $NMI(A, B) = 0$. A larger value of NMI represents a greater similarity between A and B.

Fig.2. and Fig.3. show the NMI and FVCC performances on computer-generated networks for different values of N, φ and k. For NMI performances, we can get at least about 90% of the true communities for the networks with $N = 1000$ or $= 5000$ until $\varphi = 0.4$. For FVCC performances, we can get 100%until $\varphi = 0.4$. FVCC takes the largest module that is found in true communities as a correct identification, which will take the subdivision of the network structure as the right one, and our algorithm subdivides the network that leads to a lower NMI. Fig.4. and Fig.5. show the performances of ACCS with different degree, and from the figures we can see that the ACCS would have high efficiency with the degree increases, that means the algorithm would have high efficiency with the

TABLE I. REAL-WORLD NETWORKS USED IN OUR EXPERIMENT

Networks	V(G)	E(G)	Description
CA-GrQc	5242	28980	General Relativity and Quantum Cosmolog collaboration network [15]
Word	7207	31784	Semantic network [12]
Hep-th	8361	15751	High-Energy Theory collaboration Network [13]
CA-HepTh	9877	51971	High Energy Physics - Theory collaboration network [11]
Astro-ph	16706	121251	Astrophysics collaboration Network [13]
Internet	22963	48436	A Snapshot of the Internet by Mark Newman [14]
Email-Enron	36692	367662	Enron email communication network [15]

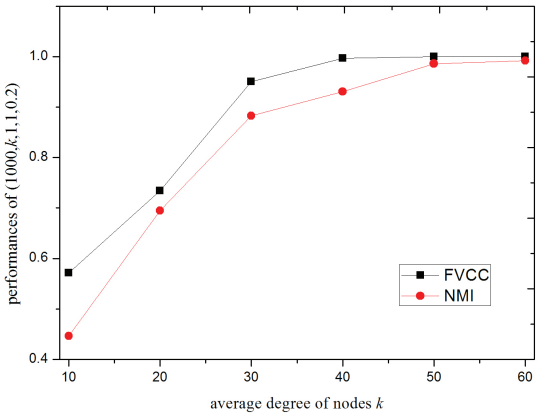


Fig. 4. The (1000 k 1 1 0 2) performances

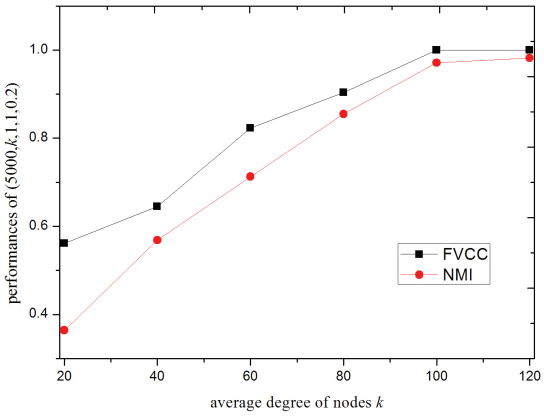


Fig. 5. The (5000,k,1,1,0.2) performances

networks connected tightenly. On the whole, the ACCS is still able to effectively identify the communities in computer-generated networks.

B. Real-world Networks

As real-world networks may have some different topological properties from the computer-generated ones, here we adopt several widely used large-scale real-world networks to further evaluate the performance of our algorithm. The networks that we use are listed in Table 1. The smallest of these networks has 5242 nodes, and 28980 edges, while the largest one has 36692 nodes, and 367662 edges. These networks include the scientists collaboration network, the email network, the word network and the internet network, which are provided by Newman, Leskovec and Palla.

The community structure of the real-world networks is always unknown, so we employ the most commonly used module function Q [1] to evaluate the quality of our algorithm. Table 2 shows the performance comparison between ACCS and FN after many experiments.

As we can see from Table 2, the run-time of ACCS is less than FN and ACC-FP for all different sampling rates and different networks, and the result of Q-value is complicated, the concrete analysis as follows:

(1) The comparison between FN and ACC-FP: In network Word, the Q-value of ACC-FP is 0.4020, which is better than FN. In network CA-GrQc, Hep-th and CA-HepTh, the Q-value of ACC-FP is 0.5242, 0.5213 and 0.4436 respectively, which are all less than FN. In the network Astro-ph, Internet and Email-Enron, ACC-FP can not get result due to the insufficient memory, and FN can get 0.3923, 0.4097 and 0.3742 respectively.

(2) The comparison between FN and ACCS: the Q-value of ACCS is better than FN when $R = 0.2$ while inferior to FN when $R = 0.1$ and 0.15, Both the network CA-GrQc and Internet belong to this case. The Q-value of network Word in ACCS is better than FN from $R = 0.15$ while inferior to FN when $R = 0.1$. The Q-value of ACCS is inferior to FN when $R = 0.1$, 0.15, and 0.2, and the network Hep-th, CA-Hepth, Astro-ph, Email-Enron belong to this case.

(3) The comparison between ACC-FP and ACCS: In network CA-GrQc, the Q-value of ACCS is 0.5253, which is better than ACC-FP. In network Word, Hep-th and CA-HepTh, the Q-values of ACCS are all less than ACC-FP. And in the rest of the networks, the ACCS can also get good results while the ACC-FP cannot.

In summary, the Q-value in our algorithm is comparable with that in FN and ACC-FP, but the run-time is shorter than FN and ACC-FP, especially with the increase of the network size. FN takes modularity Q as the objective function, when a new node joins the community, the algorithm need to compute the value of Q which is time-consuming. In our algorithm, sampling strategy reduces the scale of the network, and computing the value of Q used in the community merging phase lowers the computational complexity which compared

TABLE II. COMPARE SACC AND FN IN TERMS OF Q-VALUE AND RUN-TIME

Networks	Q-value/Run-Time(s)				
	FN	ACC-FP	ACCS ($R = 0\ 1$)	ACCS ($R = 0\ 15$)	ACCS ($R = 0\ 2$)
CA-GrQc	0.5250/44.37	0.5242/3820.21	0.4927/29.83	0.5187/38.47	0.5253/43.20
Word	0.3821/48.25	0.4020/4131.00	0.3722/32.70	0.3840/38.84	0.4061/46.21
Hep-th	0.5560/51.83	0.5213/2790.44	0.3923/35.08	0.4907/44.13	0.5171/49.22
CA-HepTh	0.4457/82.48	0.4436/5050.75	0.4043/46.56	0.4106/53.52	0.4355/59.58
Astro-ph	0.3923/190.69	-	0.3702/86.94	0.3743/98.86	0.3811/110.08
Internet	0.4097/370.48	-	0.3820/187.32	0.4002/213.02	0.4134/259.10
Email-Enron	0.3742/760.75	-	0.2948/230.58	0.3067/270.33	0.3594/402.43

with FN. On the other hand, the bigger the sampling rate is, the more information we can gain from the original network, and the bigger value of Q we can get, but meanwhile the longer run-time the algorithm consumes, so the sampling rate R is a balance between the Q-value and the run-time in our algorithm.

On both the real-world networks or on the computer-generated networks, the results obtained show the capability of our algorithm effectively deals with community detection in large-scale networks.

IV. CONCLUSIONS

The rapid development of social media platform provides us a lot of real-world social networks, which makes us quickly enter the era of big data, meanwhile, taking the reason that reveal unknown relationships between nodes and provide useful information for unknown nodes into account, more and more researchers focus on the community detection of large-scale complex networks. In this paper, an ant colony clustering algorithm based on sampling to detect community in large-scale complex networks has been proposed. The algorithm employs a sampling method, which greatly reduces the complexity of the algorithm. And it uses the idea of "label propagation" in the clustering phase, so as to distinguish which community the node belongs to. Moreover, we propose a new similarity metric between nodes and communities to obtain a partition of the initial network. Experimental results confirm that ACCS can not only greatly improve the speed of detecting community, but also has the ability to balance between getting high quality solutions and operation efficiency. Future research will aim at making "clustering the sampled nodes" and "assigning the unsampled nodes" synchronized to improve quality results.

ACKNOWLEDGMENT

This work is partly supported by National "973" Key Basic Research Program of China (2014CB744601), NSFC Research Program (61375059, 61332016), Specialized Research Fund for the Doctoral Program of Higher Education (20121103110031), and the Beijing Municipal Education Research Plan key project (Beijing Municipal Fund Class B) (KZ201410005004).

REFERENCES

[1] Newman MEJ, *Fast Algorithm for Detecting Community Structure in Networks*. Physical Review E, vol.69 (6), 066133, 2004.

[2] He D X, Liu J, Liu D Y, Jin D and Jia Z X, *Ant Colony Optimization for Community Detection in Large-scale Complex Networks*. 2011 Seventh International Conference on Natural Computation, pp.1151-1155, 2011.

[3] Jin D, Yang B, He D X and Liu D Y, *Genetic algorithm with local search for Community Detection in Large-scale Complex Networks*. Acta Automatica Sinica, vol. 37, no. 7, pp. 873-882, 2011.

[4] U. N. Raghavan, R. Albert and S. Kumara, *Near Linear Time Algorithm to Detect Community Structures in Large-scale Networks*. Physical Review E, vol.76, no. 3, 2007.

[5] Zhou S G, Zhou A Y, Jin W, Fan Y and Qian W N, *DBSCAN: A Fast DBSCAN Algorithm*. Journal of software, vol.11. no. 6, pp. 735-744, 2000.

[6] Guha S, Rastogi R and Shim k, *CURE: An Efficient Clustering Algorithm for Large Databases*. Information Systems, vol.26, no. 1, pp. 35-58, 2001.

[7] Ji J Z, Song X J, Liu C N and Zhang X Z, *Ant Colony Clustering with Fitness Perception and Pheromone Diffusion for Community Detection in Complex Networks*. Physica A, vol. 392, no. 15, pp. 3260-3272, 2013.

[8] Li K, Gong X F, Guan S G and Lai CH, *Efficient Algorithm Based on Neighborhood Overlap for Community Identification in Complex Networks*. Physica A, vol. 391, no. 4, pp. 1788-1796, 2012.

[9] Lancichinetti A, Fortunato S and Radicchi F, *Benchmark Graphs for Testing Community Detection Algorithms*. Physical Review E, vol. 78, no. 4, 046110, 2008.

[10] Danon L, Diaz-Guilera A, Duch J and Arenas A, *Comparing community structure identification*. Journal of Statistical Mechanics-Theory and Experiment, vol. 78, no. 9, P09008, 2005.

[11] J. Leskovec, J. Kleinberg and C. Faloutsos, *Graph Evolution: Densification and Shrinking Diameters*. ACM Transactions on Knowledge Discovery from Data (ACM TKDD), vol. 1, no. 1, 2007.

[12] G. Palla, I. Derenyi, I. Farkas, and T. Vicsek, *Uncovering the Overlapping Community Structures of Complex Networks in Nature and Society*. Nature, vol. 435, pp. 814-818, 2005.

[13] Newman M E J, *The structure of scientific collaboration networks*. Proceedings of the National Academy of Sciences, vol. 98, no. 2, pp. 404-409, 2001.

[14] Network data from Mark Newman's home page, http://www-personal.umich.edu/$\sim$ mejn/net data/, 2006.

[15] J. Leskovec, K. Lang, A. Dasgupta, M. Mahoney, *Community Structure in Large Networks: Natural Cluster Sizes and the Absence of Large Well-Defined Clusters*. Internet Mathematics, vol. 6, no. 1, pp. 29-123, 2009.