Knowledge-Based Systems xxx (2013) xxx–xxx

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

A fast parallel modularity optimization algorithm (FPMQA) for community detection in online social network

Zhan Bu [a,b], Chengcui Zhang [b], Zhengyou Xia [a], Jiandong Wang [a]

[a] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China
[b] Computer and Information Sciences, The University of Alabama at Birmingham, USA

ARTICLE INFO

Article history:
Received 7 December 2012
Received in revised form 27 May 2013
Accepted 19 June 2013
Available online xxxx

Keywords:
Online social network
Interest Network
Similar-View Network
Community detection
Parallel

ABSTRACT

As information technology has advanced, people are turning more frequently to electronic media for communication, and social relationships are increasingly found in online channels. Discovering the latent communities therein is a useful way to better understand the properties of a virtual social network. Traditional community-detection tasks only consider the structural characteristics of a social organization, but more information about nodes and edges such as semantic information cannot be exploited. What is more, the typical size of virtual spaces is now counted in millions, if not billions, of nodes and edges, most existing algorithms are incapable to analyze such large scale dense networks.

In this paper, we first introduce an interesting social network model (Interest Network) in which links between two IDs are built if they both participate to the discussions about one or more topics/stories. In this case, we say both of the connected two IDs have the similar interests. Then, the edges of the initial network are updated using the attitude consistency information of the connected ID pairs. For a given ID pair i and j, they may together reply to some topics/IDs. The implicit orientations/attitudes of these two IDs to their together-reply topics/IDs may not be the same. We use a simple statistical method to calculate the attitude consistency, the value of which is between 0 and 1, and the higher value corresponds to a greater degree of consistency of the given ID pair to topics/IDs. The updated network is called Similar-View Network (SVN). In the second part, a fast parallel modularity optimization algorithm (FPMQA) that performs the analogous greedy optimization as CNM and FUC is used to conduct community discovering. By using the parallel manner and sophisticated data structures, its running time is essentially fast, $O(k^{\max}(k^{\max} + \langle k \rangle \log k^{\max}))$. Finally, we propose an evaluation metric, which is based on the reliable ground truths, for online network community detection. In the experimental work, we evaluate our method using real datasets and compare our approach with several previous methods; the results show that our method is more effective and accurate in find potential online communities.

© 2013 Elsevier B.V. All rights reserved.

1. Introduction

Recently, online social networks [17,15] have gained significant popularity and are now among the most popular sites on the Web. For example, Tianya (over 32 million users), MySpace (over 190 million), Google+ (over 400 million), and Facebook (over 800 million) are popular sites built on social networks. Unlike the Web, which is largely organized around content, online social networks are organized around users [3]. Participating users join a network, publish their profile (optionally) and any content. And other users comment on this content so as to continue the discussion. The resulting social network provides a platform for maintaining social relationships, finding users with similar interests, and locating content and knowledge that has been contributed or endorsed by other users. Users with the similar interests can join the same community or group. When compared to the rest of the network, distinct communities or groups within networks can loosely be defined as subsets of nodes which are more densely linked. Understanding the structure and dynamics of social groups is a natural goal for network analysis, since such groups tend to be embedded within larger social network structures.

A large number of researches have been devoted to the task of defining and identifying communities in social and information networks. Most previous papers on the subject of community detection in large networks noted that it is a matter of common experience that communities exist in such networks. These papers then argued that, although there is no agreed-upon definition for a community, a community should be thought of as a set of nodes that has more and/or better-connected edges between its members

Corresponding author at: College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China.
E-mail addresses: buzhan@nuaa.edu.cn, zhanb@cis.uab.edu (Z. Bu), zhang@cis.uab.edu (C. Zhang), zhengyou_xia@nuaa.edu.cn (Z. Xia), aics@nuaa.edu.cn (J. Wang).

0950-7051/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.knosys.2013.06.014

ARTICLE IN PRESS

than between its members and the remainder of the network. These papers then apply a range of algorithmic techniques and intuitions to extract subsets of nodes and then interpret these subsets as meaningful communities. These algorithms focus on optimizing an energy-based cost function that is always defined with fixed parameters over possible community assignments of nodes. A notable work proposed by Newman and Girvan [7] introduced modularity as a posterior measure of network structure. This metric has been influential in the community-detection literature and has found success in many applications [28,19,25,29,26,1].

As members in the same online community may have common hobbies, social functions, occupations, interests on some topics, viewpoints, etc., they are more likely to appear in the same posts or comment on the same topics with similar perspectives. Most previous papers [28,19,25,29,2] on the subject of community detection mainly focus on link analysis or topological structure of the network; the relationship between two IDs is often directly measured according to their interactive times. In fact, online social network contains rich textual data which can be used to measure the relationship between two connected IDs. For example, we can learn the implicit orientation from one ID to the other by analyzing the emotional words appeared in their comments. Fig. 1(a) shows a tree structure corresponding to a small thread of depth 4. Labels denote the user who writes the contribution and valid comments are shown within the gray region. The post triggers three responses from users A, C and D. At the second nesting level, eight comments appear. At the third level, there are seven comments and finally, there is one last comment from C. The attitude of every comment can be represented using + or −, with + denoting a user is supportive to the viewpoint and – otherwise. Fig. 1(b) is a social network excavated from original thread of comments, including seven nodes and nine edges. The nodes represent the members involved in the social activities and the edges represent the social relations of interactions or communications. The weight attached to each edge represents the strength of connections between the corresponding members. Fig. 1(c) shows the result of discovered communities based on link analysis. We can see that members within a community are connected, but their opinions are different. In the left community of Fig. 1(c), user B is strongly opposed to the viewpoints of user A, in reality, these two IDs should not come from the same online community. Fig. 1(d) shows an ideal result, the members within one community not only have the same interests (comment to the same topics/IDs together), but also have consistent perspectives to a certain topic. Existing studies on the subject of community detection may confuse the meanings of online community, which needs to improve.

What is more, the typical size of virtual spaces is now counted in millions, if not billions, of nodes and edges. Most existing algorithms [7,22,6,23,14,10,11,9] are incapable to analyze large-scale dense networks. In our previous work [26], the fastest approximation algorithm for optimizing modularity on large networks (CNM algorithm [7]) was applied to analyze various subsets of a comment relationship network obtained from a bulletin board system (BBS). CNM consists in recurrently merging communities that optimize the production of modularity. It uses a clever data structure to store and retrieve information required to update matrix Q. As the community merge process is always serial, GNM runs well only for a mid-scale subset of the network, it is incapable to analyze larger networks. In the case of a sample graph as shown in Fig. 2(a), GNM begins with each node as a separate community in a network. Then, it finds the pair of communities with the global maximum ΔQ, and merges the pair into one community. For instance, in the initial graph, the global maximum ΔQ is 0.051, which is contributed by communities C_0 and C_1. We merge this community pair into one community as C_1', at the same time, the ΔQ on corresponding community pairs will be updated. The CNM algorithm continues the second step until the global maximum ΔQ is not positive any more, as shown in Fig. 2(d)–(f). In fact, in the first step of CNM, we can also merge communities C_4 and C_5. The two merge processes can be parallel performed as shown in Fig. 2(g). In the

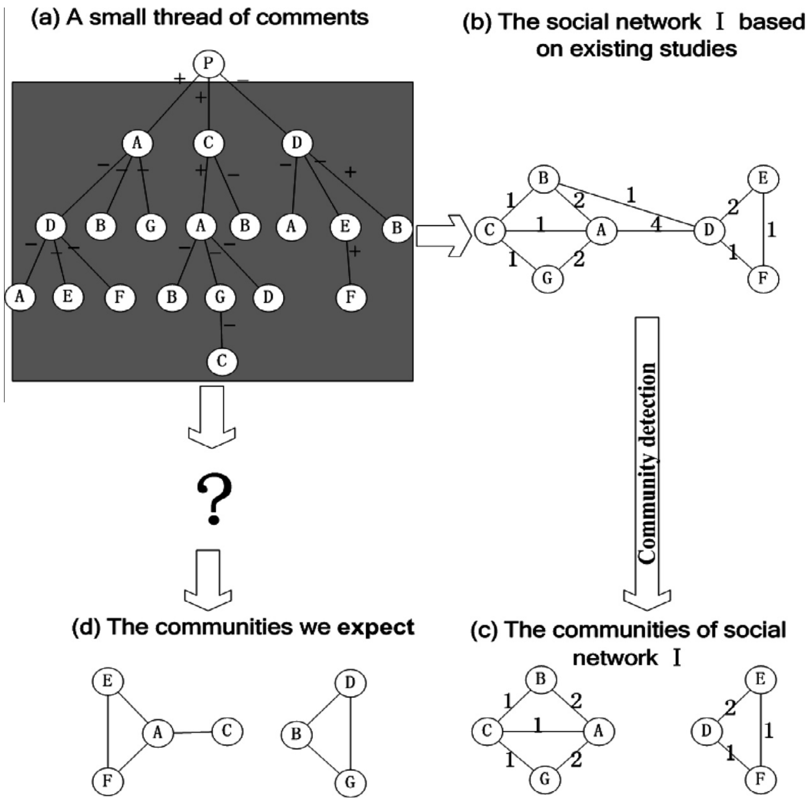
(a) A small thread of comments
(b) The social network Ⅰ based on existing studies
Community detection
?
(d) The communities we expect
(c) The communities of social network Ⅰ
P
A
B
C
D
E
F
G
+
−
1
2
4

Fig. 1. An example to illustrate the practice meaning of online community.

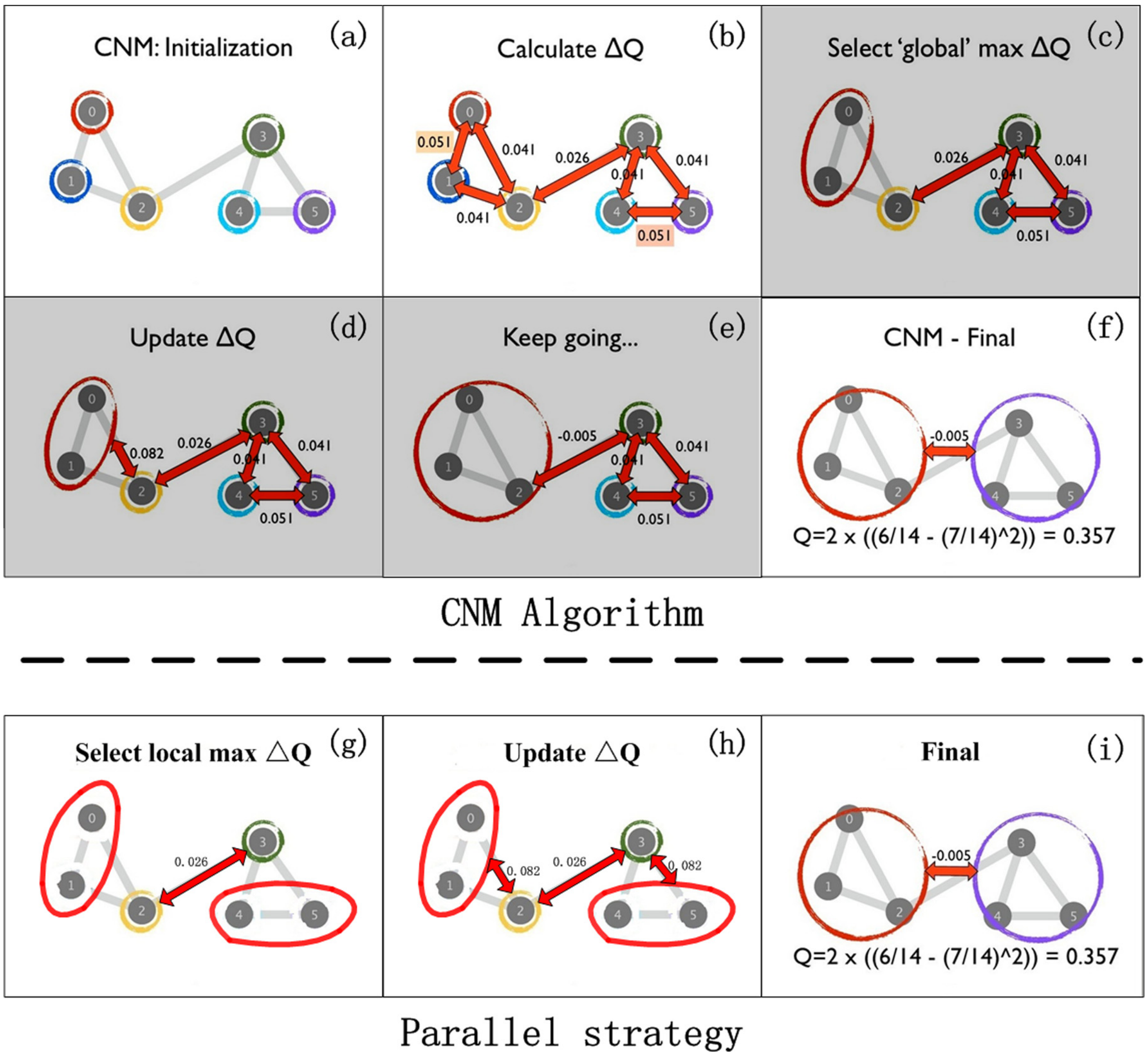

Fig. 2. An example to illustrate the parallel strategy.

second step, we can also parallel merge community pairs (C'_1, C_3) and (C_4, C'_5) (see Fig. 2(i)). It takes four steps for CNM to get the final result. However, by taking parallel strategy, the algorithm needs only two steps. Therefore, there is a great improvement space for the initial CNM algorithm.

The third problem lies in community detection is the lack of reliable ground truth. Currently the performances of community detection methods are evaluated by manual inspection. For each detected community an effort is made to interpret it as a "real" community by identifying a common property or external attribute shared by all the members of the community. For example, given a scientific collaboration network we identify communities based on the structure of the network and then find that these communities correspond to real scientific organizations. Thus, the goal of community detection is to identify sets of nodes with a common (often external/latent) function based only the connectivity structure of the network. A common function can be common role, affiliation, or attribute [27]. In our scientific collaboration network example above, such common function of nodes would be "working in common areas of science". However, such anecdotal evaluation procedures require extensive manual effort, are non-comprehensive and limited to small networks. What is more, in online social networks, those common functions are likely incomplete and missing. Not every user in online social networks is willing to provide their true and complete personal information. Therefore, we need a gold-standard metric to evaluate the accuracy of detected communities.

The contributions of our work are threefold. We first introduce an interesting social network model (Interest Network) in which links between two IDs are built if they both have participated to the discussions about one or more topics/users. In this case, we say both of the connected two IDs have the similar interests. The edges of the initial network were updated using the attitude consistency of the connected ID pairs. For a given ID pair *i* and *j*, they may together reply to some topics/users. The implicit orientations/attitudes of these two IDs to their together-reply topics/users may not be the same. We use a simple statistical method to calculate the attitude consistency, the value of which is between 0 and 1, and the higher value corresponds to a greater degree of consistency of the given ID pair to topics/users. The updated network is called Similar-View Network (SVN). Then, a fast parallel modularity optimization algorithm (FPMQA) that performs the analogous greedy optimization as CNM [7] and FUC [6] is used to conduct community discovering. FPMQA begins with each node as a separate community in a network. Then, in every step, it finds the pairs of communities with the local maximum ΔQ, only if it is positive, and parallel merges the corresponding community pairs into one community. The FPMQA continues until there are

ARTICLE IN PRESS

no more changes. By using the parallel manner and sophisticated data structures, its running time is essentially fast, $O(k^{\max}(k^{\max} + \langle k \rangle \log k^{\max}))$. Finally, we propose an evaluation metric, which is based on the reliable ground truths, for online network community detection. In the experimental work, we evaluate our method using real datasets and compare our approach with several previous methods using this metric; the results show that our method is more effective and accurate in find potential online communities.

The remainder of this paper is organized as follows. Section 2 introduces the motivation and dataset. The Interest Network (IN) model and Similar-View Network (SVN) model are presented in Section 3. In section 4, several characteristics of IN and SVN are identified through statistical analysis. In Section 5, a fast parallel modularity optimization algorithm (FPMQA) is particularly studied, followed by experiments in Section 6. Finally, we conclude the paper in Section 7.

2. Motivation and dataset

In this section, the research motivation and the data we used are introduced.

2.1. Motivation

Discovering the latent communities is a most important research problem in the social network. However, traditional researches on the subject of community detection in this field have three shortcomings: (1) they confuse the meanings of the online communities in social networks. Most of previous works (e.g., [28,19,25,29] focus on link analysis or topological structure of the network; the relationship between two IDs is often directly measured according to their interactive times. In fact, online social network contains rich textual data which can be used to measure the relationship between two connected IDs. (2) Most community detection algorithms [7,22,6,23,14,10,11,9] are incapable to analyze large-scale dense networks. (3) Finally, there is short of the reliable ground truth to evaluate the results of community detection. In this paper, we will propose some innovative technologies to solve the above three problems.

2.2. Dataset

Our real world data used in this paper is download form Tianya forum (http://focus.tianya.cn). It is a popular bulletin-board service in China. It includes more than 300 boards, and the total number of registered user identifications (IDs) is more than 32 million. Since its introduction in 1999, it has become the leading social-networking site in China due to its openness and freedom. We selected the worldview board and collected data between July, 2003 and December, 2011. The data includes 324,666 users, 99,753 topics and 4,712,859 comments. We also use a number of test-case networks (e.g., Karate [30], Facebook [18], LiveJournal [27]) to evaluate the effectiveness of our proposed fast parallel modularity optimization algorithm.

3. Network model

We first review some related works on online community detection, and then we propose an interesting network model (Interest Network) in which links between two IDs are built if they both have participated to the discussions about one or more topics/stories. Finally, we apply sentiment analysis to every comment and update the initial Interest Network using the attitude consistency information of the connected ID pairs. The updated network is called Similar-View Network.

3.1. Related work on online community detection

Online social network (e.g., Tianya, MySpace, Google+, and Facebook) are an appealing way for members of such group to communicate because they are easily accessed from almost anywhere in the world. Recent applications of data mining to online social networks have shown that increasing amounts of real data are network structured [13,16]: the users in these networks (people, IP addresses, etc.) are usually modeled by nodes of graphs; the connection relations (trust or dependent relations) between members are represented by the graph edges. Though such data often involve massive relational information among objects, some researchers attempt to find potential communities or groups in online social networks [28,19,25,29,26].

A framework for user activity analysis on an interactive website is proposed by Zeng et al. in [28]. Their approach, modeling user activity as a hidden Markov model (HMM), can be apply to user interest computation and user activity analysis tasks. To discover the discussion topics of social networks, McCallum et al. present the Author–Recipient–Topic model [19]. The model builds on Latent Dirichlet Allocation (LDA) and the Author–Topic (AT) model, adding the key attribute that distribution over topics is conditioned distinctly on both the sender and recipient—steering the discovery of topics according to the relationships between people. Tian et al. propose OLAP-style aggregation strategies to partition the graph according to attribute similarity, so that nodes within one community share the same attribute values [25]. Zhang et al. propose a topic oriented community detection approach which combines both social objects clustering and link analysis [29]. They first use a subspace clustering algorithm to group all the social objects into topics. Then they divide the members that are involved in those social objects into topical clusters, each corresponding to a distinct topic. In order to differentiate the strength of connections, they perform a link analysis on each topical cluster to detect the topical communities. In our most recent work [26], we learn the implicit orientation from one ID to the other by analyzing the emotional words appeared in their comments. Then, we build a semantic network model using those learned orientation between two users. Community detection on this semantic network performs well in effective and speed.

The above methods aim to group members who frequently communicate with each other into one community. However, they confuse the meaning of online community. In reality, members in one community may not need to communicate with each frequently, but should have similar interests and consistent perspectives to most topics they participate together. To address this problem, we first introduce an interesting social network model (Interest Network) in which links between two IDs are built if they both have participated to the discussions about one or more topics/stories. Then, we apply sentiment analysis to every comment and update the initial Interest Network using the attitude consistency information of the connected ID pairs.

3.2. Interest Network

In general, a network is built according to the implicit relations between the author of a comment and the user who replies to it. To improve the quality of the resulting network, some of the comments need filtering according to the following criteria:

(1) Those self-replies should also be filtered.
(2) Anonymous comments will not be reserved.

In a social network, every registered user identification (ID) corresponds to a node $i \in V$ in a graph $G = \langle V, E \rangle$. An edge $(i, j) \in E$ represents a social relation between two users that results from their comment activity. Let n_{ij} be the number of times that user i writes a comment to user j, an undirected edge exists between users i and j if $n_{ip} > 0$ and $n_{jp} > 0$ where $i \neq p, j \neq p$. The weight on the edge (i, j) is defined as $w_{ij} = \sum_{p \in P_{ij}} \min(n_{ip}, n_{jp})$, where P_{ij} is the set of users to whom user i and j together comment. The network established according to this way is called Interest Network (IN). Fig. 3 gives a simple example to IN model. Take edge (A, E) as an example, user A and user E comment to user D twice together. One time is in the second nesting level, and the other is in the third level. As $n_{AD} = n_{ED} = 2$, the weight on the edge (A, E) is 2. This edge-build process is repeated for each node pair, and the final Interest Network is shown in Fig. 3(b).

3.3. Similar-View Network

We can infer that the connected users in Interest Network have similar interests, because they comment to the same users together. However, their viewpoints to the same topics may not be the same. Take the user pair (A, B) as an example, user A is supportive to the viewpoint of user C's and opposes user D's. While user B takes the opposed action, s/he opposes user C and supports user D. Their attitudes to the same topics/users are totally opposite. However, the weight on edge (A, B) is 2 in Interest Network, which is very high. Therefore, there is a need to re-measure the weight of edge in Interest Network.

Online comments serve as a simple and effective way for users to interact with their readership. They are among the defining set of weblog characteristics, and most posters identify comment feedback as an important motivation for their writing. What's more, on examining every comment, we may find that most of them have an implicit orientation that is mostly appraised by several emotional words. Those emotional words basically include two types: supportive and opposing. For example, phrases such as "顶", "经典" or "牛" are supportive, whereas words such as "NND", "TM" or "白痴" are opposing. We count emotional terms/phrases in the comment data, including both supportive words and opposing ones; and select 50 items with maximum frequency respectively; then every term/phrase is assigned with a value between 0 and 1 according to their tone manually. A higher value corresponds to a greater degree of support; if the phrase is neutral, we assigned it a value of 0.5. Thus, every phrase has an associated numerical "trust". In Table 1 we roughly identified several terms or phrases, with English version in parentheses, from a public discussion on Tianya.com as either supportive or opposing. Accordingly, every comment between two users is analyzed and the "trust" value of every comment is updated (if there are several emotional words in one comment, we take the average). As shown in Fig. 3(a), the comment with the "trust" value higher than 0.5 will be defined as positive comment, the comment with the "trust" value lower than 0.5 will be defined as negative one. In this paper, we do not consider those neutral comments. Accordingly, the "trust" from user i to user p about the topic t can be calculated as:

Table 1
Emotional phrases with English version in parentheses associated with scores.

	Phrases	Core	Orientation
1	顶/ding (Support)	1.0	Supportive
2	经典 (Classic)	0.8	Supportive
3	沙发 (First to reply)	0.7	Supportive
4	牛 (Fantastic)	0.7	Supportive
5	喜欢 (Like)	0.7	Supportive
...	...	...	...
...	...	...	...
46	NND/nnd (TNND)	0.25	Opposing
47	SB/sb (Shithead)	0.1	Opposing
48	TM/tm (Fuck)	0.25	Opposing
49	YY/yy (Psychosexuality)	0	Opposing
50	白痴 (Idiot)	0.2	Opposing

$$\text{trust}_{ip}^{t} = \frac{\sum_{c \in C_{ip}^{t}} turst(c_{ip}^{t})}{n_{ip}^{t}} \tag{1}$$

where $turst(c_{ip}^{t})$ is the "trust" value of one comment from user i to user p about the topic t, C_{ip}^{t} is the comment set which includes all the comments from user i to user p about the topic t, and n_{ip}^{t} is the comment number from user i to user p about the topic t.

For every connected user pair in Interest Network, we introduce a parameter to measure the attitude consistency to their together-reply topics/users. The attitude consistency of user i and j, AC_{ij} is defined as:

$$AC_{ij} = \frac{\sum_{t \in T_{ij}} \sum_{p \in P_{ij}^{t}} \delta(\text{trust}_{ip}^{t}, \text{trust}_{jp}^{t})}{\sum_{t \in T_{ij}} count(P_{ij}^{t})} \tag{2}$$

where T_{ij} is a topic set, it includes the topics which are together discussed by user i and j. P_{ij}^{t} is a user set, it includes the users to whom user i and j together comment in the discussion of topic t. $count(P_{ij}^{t})$ is the size of the user set P_{ij}^{t}. And $\delta(x, y)$ is a judgment function determined by x and y, which obeys:

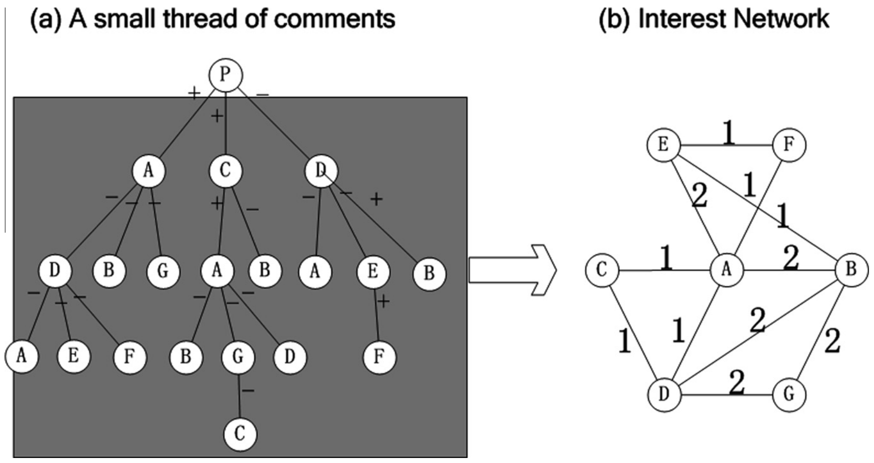

Fig. 3. An example to illustrate the Interest Network model.

$$\delta(x,y)=\begin{cases}1 & x>0.5,\ y>0.5\ \ or\ \ x<0.5,\ y<0.5\\ 0 & otherwise\end{cases}$$

The range of AC_{ij} is between 0 and 1, and a higher value corresponds to a greater degree of attitude consistency of the given user pair to their together-reply topics/users. Then, we can update the weight on every edge in Interest Network using this AC_{ij}, as shown in Fig. 4(b). Take user pair (A, B) in Fig. 3(a) as an example. User A is supportive to the viewpoint of user C, then $\text{trust}^t_{AC}=\frac{\sum_{c\in C^t_{AC}} turst(c^t_{AC})}{n^t_{AC}}>0.5$. While user B opposes user C's viewpoint, $\text{trust}^t_{BC}=\frac{\sum_{c\in C^t_{BC}} turst(c^t_{BC})}{n^t_{BC}}<0.5$. Similarly, we can get $\text{trust}^t_{AD}<0.5$ and $\text{trust}^t_{BD}>0.5$. Therefore, the attitude consistency of user A and B is calculated as:

$$\begin{aligned}AC_{AB}&=\frac{\sum_{t\in T_{AB}}\sum_{p\in P^t_{AB}}\delta(\text{trust}^t_{Ap},\text{trust}^t_{Bp})}{\sum_{t\in T_{AB}} count(\mathrm{P}^t_{AB})}\\&=\frac{\delta(\text{trust}^t_{AC},\text{trust}^t_{BC})+\delta(\text{trust}^t_{AD},\text{trust}^t_{BD})}{2}=\frac{0+0}{2}=0\end{aligned}\qquad(4)$$

This updating process is repeated for each edge in initial Interest Network. Finally, we prune the edges on which AC_{ij} equals 0 (see Fig. 4(c)). The pruned IN can be called Similar-View Network (SVN).

4. Statistical analysis of the Similar-View Network

In this section, the statistical properties of the Similar-View Network are analyzed, and we compare the results with some existing models to characterize how they differ or resemble one another.

4.1. Global properties

Here, we discuss network characteristics from a global perspective. The detailed statistics of the Similar-View Network are listed in Table 2 along with those of the other three networks. The connectivity of the network (row 3) is the ratio of actual links m to the potential number of links $O(n^2)$ (The number of the edges of a complete graph.) As shown in Table 2, SVN is highly dense compared to the others. In SVN, the "giant component" comprises 86.13% of the users. In SVN, $\langle k\rangle$ is high, meaning that users in this network have a relatively large circle of friends and resemble each other in their interests. The maximum degrees of the four networks are shown in row 6. The clustering coefficient of a node i is defined as $c'_i=\frac{2e_i}{k_i(k_i-1)}=\frac{\sum_{j,m}a_{ij}a_{jm}a_{mi}}{k_i(k_i-1)}$, where $a_{jm}=1$ for two neighbors j and m of node i. The clustering coefficient of the whole network is the average of the individual c'_i. We observed that, for SVN, c is much higher than the randomized counterpart which is defined as $c_{rand}=\langle k\rangle/N$. The average shortest path length is small for SVN, suggesting that it is a "small-world" network. The diameter D of this social network is also very small. This has also been seen in other traditional social networks. Another statistic of social networks is the degree correlation, or mixing coefficient, that indicates whether highly connected users are preferentially linked to other highly connected users. Table 2 shows the correlation coefficient r [20,21] (also called the Pearson correlation coefficient) for our four networks. Like traditional social networks, SVN exhibits significant assortative mixing.

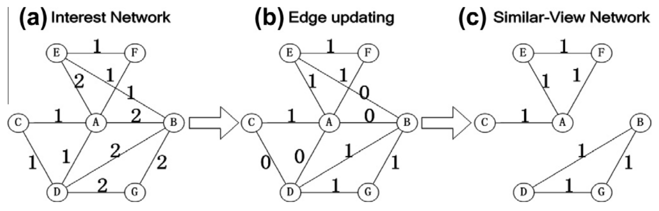

Fig. 4. An example to illustrate the Similar-View Network model.

Table 2
Statistics of the Tianya social networks. Und. Dense Netowrk (Und. DN), [13]; Semantic Network (SN), [26].

	SVN	IN	Und. DN	SN
n	154,651	318,715	323,745	162,747
m	15,323,876	40,436,642	2,987,953	678,189
Connectivity (%)	0.13	0.08	0.0057	0.0051
Maxclust (%)	86.13	90.31	75.93	69.26
$\langle k\rangle$	198.17	253.75	18.46	8.33
k^{max}	3209	5918	6505	3504
c	0.1378	0.2912	0.0712	0.0086
l	2.56	2.81	3.7781	4.2119
D	8	9	10	11
r	0.2101	0.1931	−0.0899	−0.0760

4.2. Degree distribution and others

The degree k_i of a user i, which is the number of users with whom s/he is connected, is distributed according to a power law followed by an exponential cutoff, namely, $P(k)\sim\alpha k^{\beta}$, as shown in Fig. 5(a). The cumulative distribution function (cdf) of the degrees is shown in Fig. 5(b). As expected, these distributions are all heavy-tailed, indicating a high heterogeneity between the users. The clustering function $c(k)$ is defined as the average of c'_i overall vertices with a given degree k. For the Similar-View Network and the Interest Network, $c(k)$ decays as αk^{β}, with $\alpha>0$, which is not consistent with previous reports [24]. While for the Und. Dense Network and the Semantic Network, $c(k)$ decays as $\alpha\cdot\ln(k)+\beta$, with $\alpha<0$. As shown in Fig. 5(c), the trends of the Similar-View Network and Interest Network are nearly identical. The average nearest-neighbor degree function $knn(k)$ [4,5], which is defined as the average degree of the neighbors of vertices of degree k, follows a logarithmic distribution, $knn(k)\sim\alpha\cdot\ln(k)+\beta$, for all these networks. As shown in Fig. 5(d), $knn(k)$ exhibits a slight upward curvature for the Similar-View Network. The average shortest-path degree function $l(k)$, which is defined by the average shortest path from vertices of degree k to other vertices in the "giant component", obeys a logarithmic distribution, $l(k)\sim\alpha\cdot\ln(k)+\beta$, with $\alpha<0$, meaning that hub members are more likely to be acquainted with other people. The detailed parameters of the Similar-View Network are listed in Table 3 along with those of the other three networks.

5. Community detection

Similar-View Network has vertices in a group structure, where the vertices within the group have a higher edge density, and the vertices between groups have a lower edge density. This kind of structure is called a community, which is an important network property and can reveal many hidden features of a given network. Users belonging to the same community are likely to have properties in common. Monitoring the aggregate trends and opinions revealed by these communities provides valuable insight into a number of social applications, such as criminal investigation and rumor-spreading investigations. Hence, community identification is a fundamental step not only for discovering what causes entities to form but also for understanding the overall structural and functional properties of a network. However, the typical size of virtual spaces is now counted in millions, if not billions, of nodes, most existing algorithms are incapable to analyze very larger dense networks. In this section, we first introduce some existing community-detection algorithms; then, we propose a fast parallel

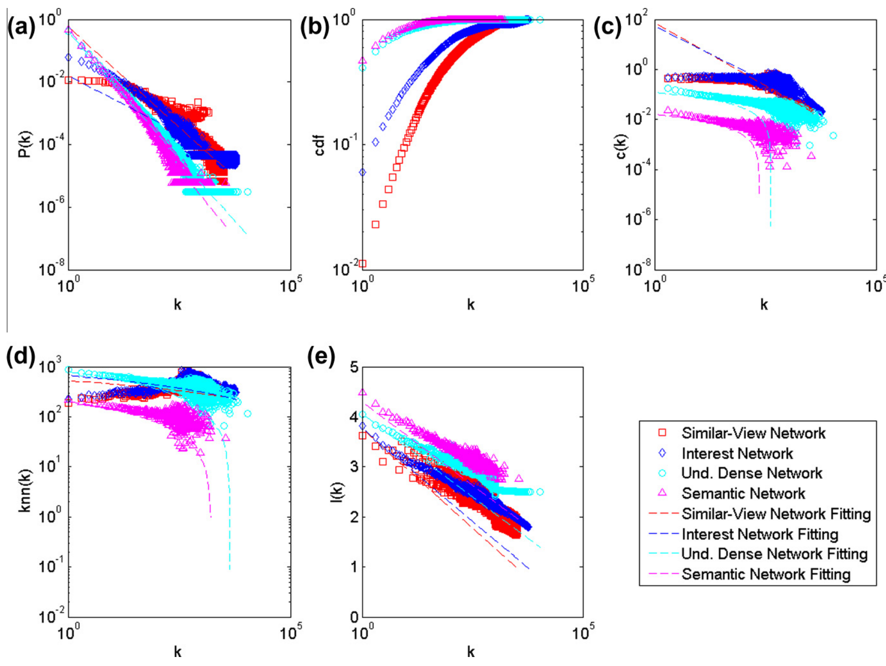

Fig. 5. P(k), cdf, c(k), knn(k), l(k) of the four networks.

Table 3
Descriptive coefficients.

	SVN	IN	Und. DN	SN
$\alpha_{P(k)}$	0.579	0.015	0.417	0.528
$\beta_{P(k)}$	−1.275	−0.716	−1.617	−1.788
$\alpha_{c(k)}$	73.052	53.305	−0.014	−0.002
$\beta_{c(k)}$	−0.998	−0.879	0.122	0.157
$\alpha_{knn(k)}$	−23.7	−35.7	−66.0	−18.8
$\beta_{knn(k)}$	528.7	670.4	797.0	202.3
$\alpha_{l(k)}$	−0.238	−0.223	−0.199	−0.200
$\beta_{l(k)}$	3.761	3.770	4.060	4.299

modularity optimization algorithm (FPMQA), which can detect communities on very large dense network. Finally, we compare it with some existing algorithms, such as CNM [7], PL [22], FUC [6], and LPA [23].

5.1. Existing community-detection algorithms

A common formulation of the problem of community detection is to find a partitioning $C = \{C_1, C_2, \ldots, C_K\}$ of disjoint subsets of vertices of the graph $G = \langle V, E \rangle$ representing the network, in a meaningful manner. Several algorithms have therefore been proposed to find reasonably good partitions in a reasonably fast way. It is common to differentiate by their angles, which creates the notions of division-based, agglomeration-based, optimization-based, and label-based algorithms:

(a) Divisive algorithms detect inter-community links and remove them from the network. For example, GN algorithm [14] uses edge betweenness as a metric to identify the boundaries of communities. Though it has been applied very successfully to small-scale networks, including E-mail network [10], musician network [11], and metabolic network [8], it makes heavy demands on computational resources, running in $O(n^3)$ time on a sparse network with n nodes. The cubic complexity algorithm may not be scalable enough for the size of Online Social Networks. Some other works with this notion can be referenced, such as [12,9].

(b) Agglomerative algorithms merge similar nodes/communities recursively. Pons and Latapy propose a measure of similarities between vertices based on random walks, which can be used in an agglomerative algorithm to compute efficiently the community structure of a network [22]. The computational complexity of the Pons–Latapy algorithm is $O(n^2 \log n)$ and space $O(n^2)$ in most real-world cases.

(c) Optimization methods are based on the maximization of an objective function. The most famous optimization method is in literature [7], which begins with nodes in n different communities and group together communities which has the greatest contribution to the modularity measure Q. In effect, [7] reduce the time complexity of the algorithm to $O(md \log n)$, where m is the number of edges and d is the depth of the dendrogram obtained. FUC [6] is another heuristic method that finds partitions of a given network by maximizing the modularity measure. It iteratively repeated two phases until to no increase of modularity is possible: one where modularity is optimized by allowing only local changes of communities; one where the communities found are aggregated in order to build a new network of communities. Its complexity is linear on typical and sparse data.

(d) LPA (Label Propagation Algorithm) [23] uses only the network structure as its guide, is optimized for large-scale networks, does not follows any a priori defined objective function and does not require any prior information about the communities. In addition, this technique does not need to define in advance the number of communities present into the network or their size. Labels represent unique identifiers, assigned to each vertex of the network. Its functioning is reported as described in [23]. It also takes a near-linear time for the algorithm to run to its completion.

The quality of the partitions resulting from above methods is often measured by the so-called modularity, which the fraction of edges that fall within communities, minus the expected value of the same quantity if edges fall at random without regard for the community structure. One form is given by Newman, which is defined as [7]

ARTICLE IN PRESS

$$Q = \sum_i (e_{ii} - a_i^2) = TrE - ||E^2|| \tag{5}$$

where E is a $N \times N$ symmetric matrix whose element e_{ij} is the fraction of all edges in the network that link vertices in community C_i to vertices in community C_j, and $||E^2||$ indicates the sum of the elements of the matrix E^2. The trace of this matrix $TrE = \sum_i e_{ii}$ is the fraction of edges in the network that connect vertices in the same community, while the row (or column) sums $a_i = \sum_j e_{ij}$ give the fraction of edges that connect to vertices in community C_i. If the network is such that the probability to have an edge between two sites is the same regardless of their eventual belonging to the same community, one would have $e_{ij} = a_i a_j$.

5.2. Fast parallel modularity optimization algorithm (FPMQA)

The main idea behind our parallel clustering algorithm is following. Assume that we start with a network of n nodes. First, we assign a different community to each node of the network and initialize every community/node with unique labels. Suppose that a community C_i has neighbors $C_1^i, C_2^i, \ldots, C_{k_i}^i$, we define the "local area" of community C_i as $Local_{C_i} = \{C_i, C_1^i, C_2^i, \ldots, C_{k_i}^i\}$. Then, we evaluate the gain of modularity that would take place by merging any community pair in $Local_{C_i}$. The community pair with the maximum gain of modularity will be joined as one community (the label of one community will be replaced by the other), but only if this gain is positive. If there is no positive gain, all the community pairs in the "local area" of community C_i remain unchanged. As every merging will result the gain of modularity changing in the "local areas" of the two corresponding communities. We make use of a balanced binary tree as in [7] to keep track the maximum gain of modularity for every community. A similar work proposed by [6] indicates that the ordering of the nodes does not have a significant influence on the modularity that is obtained, while it may affect the computation time. Therefore, in this paper, we parallel apply this process for all communities and the first pass is then complete. One should also note that by taking parallel processing, the gain of modularity updating may exist access conflict. For example, for community C_i, the maximum gain of modularity in its "local area" are contributed by communities C_i and C_j. While for community C_j, the maximum gain of modularity may be contributed by community pair (C_j, C_k). We cannot merge community pairs (C_i, C_j) and (C_j, C_k) simultaneously. To address this problem, we use a mark array, which stores the current state (busy or free) of every community. To merge a given community pair C_i and C_j, we should first check their current labels and states. Only if their labels are different and all the communities in their "local areas" are free, we can take the next process. When this community pair acquires the authorization, we should then mark all the communities in the "local areas" of C_i and C_j's as busy. By the end of merging, the corresponding communities will be marked as free. We perform this process interactively until there are no more changes and a maximum of modularity is attained (see karate graph in Fig. 6). By construction, the number of meta-communities decreases at each pass, therefore most of the computing time is used in the first passes.

As described above, we start off with each vertex being the sole member of a community of one. The four data structures used in our approach are described as follow:

(1) A balanced binary tree T_i for each community C_i, which stores the gain of modularity ΔQ_{ij} for each community pair in the "local areas" of C_i. So that elements can be found or inserted in $O(\log(k_i + c_i k_i (k_i - 1)/2)) \approx O(\log k_i)$ time, where c_i' is the clustering coefficient of community C_i as defined in Section 3.1.
(2) An ordinary vector array with elements a_i.
(3) An label vector array with elements l_i.
(4) An state vector array with elements s_i.

Thus, we initially set

$$\Delta Q_{ij} = 1/2m - k_i k_j/(2m)^2, \quad if\ C_j \text{ is the neighbor of } C_i \tag{6}$$

$$a_i = k_i/2m \tag{7}$$

$$l_i = i \tag{8}$$

$$s_i = free \tag{9}$$

for each community C_i. (The weighted are a simple generalization [27].)

Our algorithm can now be defined as follow:

Algorithm 1. Fast parallel modularity optimization algorithm (FPMQA)

```
1: V: a set of vertices
2: E: a set of edges
3: $G \Leftarrow \langle V, E \rangle$
4: $C \Leftarrow \{C_i = \{v_i\} | v_i \in G(V)\}$
5: $S \Leftarrow \{s_i = free | v_i \in G(V)\}$
6: $L \Leftarrow \{l_i = i | v_i \in G(V)\}$
7: $T \Leftarrow \{T_i |$insert $\Delta Q_{xy}$ in the "local area" of $C_i$ to its balanced
   binary tree $T_i\}$
8:   **while** true **do**
9:   Merge_time=0;
10: **for** $C_k \in C$ **do**
11: Merge_process($C_k$)
12: **end for**
13: **if** Merge_time==0 **then**
14: break
15: **end if**
16: **end while**
```

The operating of Merge_process(C_k)

```
17: **while** true **do**
18: find max $\Delta Q_{xy}$ from $T_k$
19: **if** $s_j$ ==free, $C_j \in Local_{C_x} \cup Local_{C_x}$ **then**
20: **if** $l_x$ == $l_y$ **then**
21: Ending Merge_process($C_k$)
22: **end if**
23: Merge_time= Merge_time+1
24: $a'_x = a_x + a_y$
25: $l'_x = l_y + l_y$
26: $s_j = busy$, if $C_j \in Local_{C_x} \cup Local_{C_x}$
27: $C'_x \Leftarrow C_x \cup C_y$
28: $C \Leftarrow C - C_x - C_y + C'_x$
29: $N'_x \Leftarrow \{C_k | C_k \in Local_{C_x} \cup Local_{C_y}\}$
30: **for** $C_k \in N'_x$ **do**
31: $\Delta Q'_{xk} = \Delta Q'_{kx} \Leftarrow Q(G, C - C_x - C_y + C'_x) - Q(G, C)$
32: $s_k = free$
33: **end for**
34: **else**
35: wait till $s_j$ ==free, if $C_j \in Local_{C_x} \cup Local_{C_y}$
36: **end if**
37: **end while**
```

The modularity updating is the as in [7]. If we join communities C_x and C_y, the degrees (the numbers of neighboring communities) of C_x and C_y can be denoted as k_x and k_y. If $k_x > k_y$, we label the

ARTICLE IN PRESS

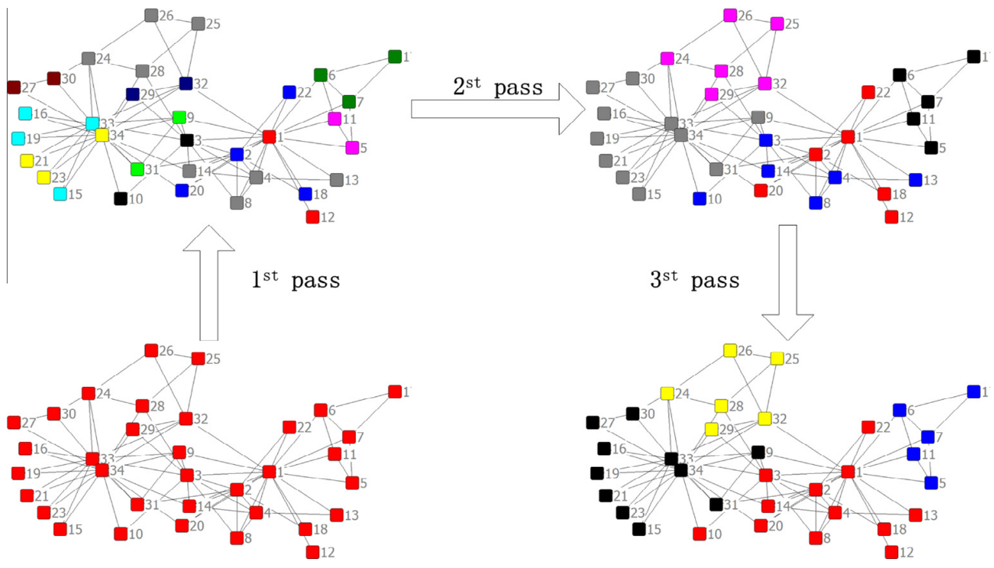

Fig. 6. Visualization of the steps of our algorithm on karate network [30]. Each pass is made of parallel merge the community pair (C_i, C_j) with the maximum gain of modularity in the "local area" of every community C_i.

combined community x, and y otherwise. The updating rules are as follows. If community C_k is connected to both C_x and C_y, then

$$\Delta Q'_{xk} = \Delta Q'_{kx} = \Delta Q_{xk} + \Delta Q_{yk} \tag{10a}$$

If C_k is connected to C_x but not to C_y, then

$$\Delta Q'_{xk} = \Delta Q'_{kx} = \Delta Q_{xk} - 2a_y a_k \tag{10b}$$

If C_k is connected to C_y but not to C_x, then

$$\Delta Q'_{xk} = \Delta Q'_{kx} = \Delta Q_{yk} - 2a_x a_k \tag{10c}$$

5.3. Algorithm analysis

We take the similar method as [7] to analyze the computational complexity of FPMQA. Take community C_i as an example. First, we need to select the community pair (C_x, C_y) with the maximum gain of modularity from C_i's balanced binary tree T_i, it will take $O(1)$ time. Then, we need to join the two communities as one. At the same time, we need update corresponding balanced binary trees. We will take a heuristic method to update the balanced binary tree T_x: (1) For every element in T_x, we give them a same decrement $(-2a_y a_k)$ without changing the tree structure to complete Eq. (10b) first. And it will take $O(k_x)$ time. (2) To implement Eq. (10a), we insert the elements of T_y into T_x, summing them wherever an community connects to both C_x and C_y. It is worth noting that we should also add a same increment $(2a_y a_k)$ to those elements, as we additionally subtract it in Step (1). Each of this *count*$(N_x \cap N_y)$ insertions takes $O(\log k_x)$ time. (3) We then update the other elements of T_x, of which there are at most *count* $(N_y - N_x \cap N_y)$, according to Eq. (10c). The total operate time for T_x is $O(k_x + k_y \log k_x)$. In the balanced binary trees T_k, we will update a single element, taking $O(\log k_k)$ times, and there are at most $k_x + k_y$ trees for us to update. Finally, the updates $a'_x = a_x + a_y$ and $l'_x = l_x + l_y$ is trivial and can be done in constant time.

By construction, the modularity updating (Steps 30–33 in FPMQA) can be also applied in parallel way. After T_k has been updated, we free the corresponding community C_k, then other communities may have the chance take the merge operating. For the pth pass, FPMQA takes $O(k_p^{\max}(k_p^{\max} + \langle k_p \rangle \log k_p^{\max}))$ time, where $k_p^{\max}$ and $\langle k_p \rangle$ are the maximum degree and the average degree of the network in the pth pass. The total time of FPMQA is $O\left(\sum_{p=1}^{d} k_p^{\max}(k_p^{\max} + \langle k_p \rangle \log k_p^{\max})\right)$, where d is the number of passes. From our experiments, we found that the main computational time comes from the first pass and 95% of the nodes or more are classified correctly by the end of the sixth pass. It is agreed with the well-known "six degree of separation" theory. However, the mathematical convergence is hard to prove. Therefore, FPMQA has a running time of $O(k^{\max}(k^{\max} + \langle k \rangle \log k^{\max}))$. Consider a star-shaped network with the center node x and leaf nodes y_i. Its average degree is about 1, and the maximum degree is n. Once a community takes the center node x to merge, we join it into x (because its degree is lower than x's). By using our heuristic method, every join will take n time as x is connected to all y_i. As every merge process will employ all node resources, other nodes will wait for the merging nodes to release resources, our method will change into a serial way. Then the overall computational time is $O(n^2)$. For a complete graph, the average degree and the maximum degree are both n, our method will also degenerate to the serial way. As every join takes $n + n\log n$ time, the overall time will be $O(n^2 \log n)$. While for other trivial sparse networks, FPMQA has a running time of $O((k^{\max})^2)$.

5.4. Comparison with other algorithms

Our algorithm also belongs to modularity optimization method. It combines the advantages of CNM [7] and FUC's [6] and has several unique features. First, it is intuitive and easy to implement, and the outcome is unsupervised. Moreover, by using clever data structures and parallel strategies, it is extremely efficient.

CNM finds the pair of communities with the "global" maximum ΔQ, however, finding this "global" maximum ΔQ is time consuming. The time complexity of CNM is $O(md \log n)$, it only runs well for a mid-scale network. Our method select the "local" maximum ΔQ as FUC does, but the definitions of the "local area" of every community are different. For each community C_i, FUC only considers the community pair between C_i and its neighboring communities. That may bring the final modularity declination in some cases. Take the graph in Fig. 7 as an example. Apparently, the graph has two communities which are {0,1,2,3} and {4,5,6}. FUC takes the nodes in a natural to start the algorithm, if we first consider node 0, its "local area" includes $\{e_{01}, e_{02}, e_{03}, e_{04}\}$. The gains of modularity on those four edges are same and equal 0.02. Then we will select a community pair randomly from its "local area". If we choose

ARTICLE IN PRESS

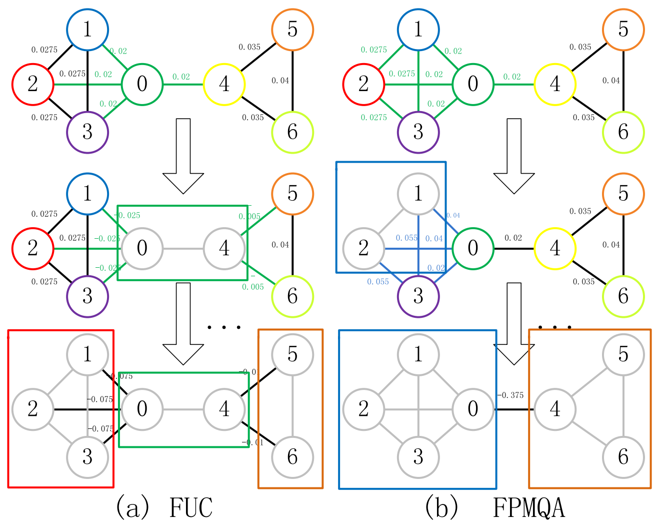

Fig. 7. Comparison of FUC and FPMQA.

community pair (0,4), we merge them as one community and update the corresponding ΔQ_{ij}. The gains of modularity of ΔQ_{01}, ΔQ_{02} and ΔQ_{03} will be updated to −0.025 according to Eq. (10), and those of ΔQ_{45} and ΔQ_{46} will be updated to −0.005. In the end of first pass, there will be only three communities left, and the algorithm will terminal as shown in Fig. 7(a). The detected communities are not same as we expect. While for our algorithm, we take the parallel strategy. Assume node 0 first acquires the needed resources, the merge process for node 0 will begin. Its "local area" includes edges between any two communities of node 0 and its neighbors, which are $\{e_{01}, e_{02}, e_{03}, e_{04}, e_{12}, e_{13}, e_{23}\}$. The maximum gain of modularity on those seven edges is 0.0275, which comes from (1,2), (1,3) or (2,3). Assume we choose community pair (1,2) to merge and update the corresponding ΔQ_{ij}. The gains of modularity of ΔQ_{01}, ΔQ_{02}, ΔQ_{13} and ΔQ_{23} will be updated to 0.04, 0.04, 0.055 and 0.055 separately. Next, we free the corresponding resources and start to consider node 1. In the end of the first pass, we detect two communities as shown in Fig. 7(b), which is our expect result. What's more, FUC considers all the nodes repeatedly and sequentially until no further improvement can be achieved. While our algorithm takes the parallel strategy, which should be faster than FUC.

We also analyze the label propagation algorithms, such as LPA in [23]. LPA initializes every node with unique labels and let the labels propagate through the network. It assumes that each node in the network chooses to join the community to which the maximum number of labels of its neighbors belongs to, with ties broken uniformly randomly. At the end of the propagation process, nodes having the same labels are grouped together as one community. Each iteration of LPA takes linear time in the number of edges $O(m)$. However, when the algorithm terminates, it is possible that two or more disconnected groups of nodes have the same label. This happens when two or more neighbors of a node receive its label and pass the labels in different directions, which ultimately lead to different communities adopting the same label. In such cases, in the end of LPA, one should run a simple breadth-first search on the sub-networks of each individual group to separate the disconnected communities with the computational time of $O(m + n)$. Therefore, the overall running time of LPA is $O(m + n)$. LPA performs better than FPMQA in those centralized networks, such as the star-shaped network, the complete network, and the networks with the maximum degree closed to n. While for other trivial networks, FPMQA is more competitive (see Table 4).

6. Experiments and evaluation metrics

In this section, the performance of FPMQA is first presented. Then, we compare the community structure of the four networks derived from Tianya data. Finally, two metrics, one based on the attitude consistency, the other based on the interest consistency, are introduced to evaluate the accuracy of our approach.

6.1. The performance of FPMQA

In order to verity the effectiveness of FPMQA, we have applied it on a number of test-case networks and compared it with four other community detection algorithms. The networks that we consider include a small social network (Karate), four subsets of online social networks (Facebook, Twitter, Gplus and Orkut), a free on-line blogging community (LiveJournal), and an on-line gaming network (Friendster). For every dataset, we first implemented FPMQA in a server with one core (in this case, FPMQA degenerates to a serial algorithm). Then we added the number of core to observe the changes of CUP time. Fig. 8 shows the CUP times and the speedup for all the test datasets. By taking the parallel strategy, FPMQA runs faster than the general serial one. One interesting finding is that the speedup of FPMQA changes with the structure of the given network. Namely, the speedup is high when we run FPMQA in a sparse network. In the case of a dense network the opposite applies. We compared FPMQA with other algorithms, the results are listed in Table 4. In all test-case networks, one can observe that the rapidity and the large values of the modularity that are obtained. FPMQA outperforms nearly all the other methods to which it compared, expect for Gplus. That is because the maximum degree of Gplus network is very high, FPMQA may degenerate to serial one. We also have applied FPMQA on the four networks derived from Tianya data. As shown in the last four rows of Table 4, even for those very

Table 4
The performances of the algorithm of PL [22], CNM [7], FUC [6], LPA [23], and our algorithm for community detection in networks of various sizes. Most of the data we used are downloaded from Stanford Large Network Dataset Collection (http://snap.stanford.edu/data/). For each method/network, the table displays the modularity that is achieved and the running time. Our method clearly performs better in terms of modularity and running time.

	Nodes/edges	$\langle k\rangle/k^{max}$	PL	CNM	FUC	LPA	FPMQA
Karate [30]	34/77	4.5/16	**0.42/0 s**	0.38/0 s	**0.42/0 s**	0.38/0 s	**0.42/0 s**
Facebook [18]	4k/88k	43.7/518	0.76/27.9 s	0.72/22 s	0.82/3.1 s	0.75/0 s	**0.84/0 s**
Twitter [18]	81k/1.8M	43.5/875	0.70/733 s	0.71/818 s	**0.73**/20.3 s	0.69/3.1 s	0.72/**1.3 s**
Gplus [18]	108k/13.7M	254.1/5126	0.66/3657 s	0.62/3321 s	0.69/69.5 s	0.65/**33.8 s**	**0.71**/45.7 s
LiveJournal [27]	4M/34.9M	17.3/1087	_/_	_/_	0.14/152.4 s	0.12/60.3 s	**0.15/22.3 s**
Orkut [27]	3M/117M	76.2/3295	_/_	_/_	0.19/437.4 s	0.18/203 s	**0.20/68.9 s**
Friendster [27]	117M/2.6B	44.4/2768	_/_	_/_	_/_	0.22/165 mn	**0.24/923.3 s**
SVN	155k/15.3M	198.2/3209	0.33/8275 s	0.35/7833 s	**0.36**/183.2 s	0.34/46.3 s	**0.36/32.7 s**
IN	319k/40.4M	253.8/5918	_/_	**0.36**/6.5 h	0.35/287.3 s	0.34/109.6 s	0.35/**89.3 s**
Und. DN [13]	324k/3.0M	18.5/6505	_/_	0.49/1731 s	**0.50**/163.2 s	**0.50/10.2 s**	**0.50**/72.8 s
SN [26]	163k/678k	8.3/3504	0.58/7535 s	0.60/350 s	**0.62**/49.4 s	0.58/**5.2 s**	**0.62**/21.2 s

Empty cells correspond to a running time over 24 h. The figures in bold means better performance among the five methods.

ARTICLE IN PRESS

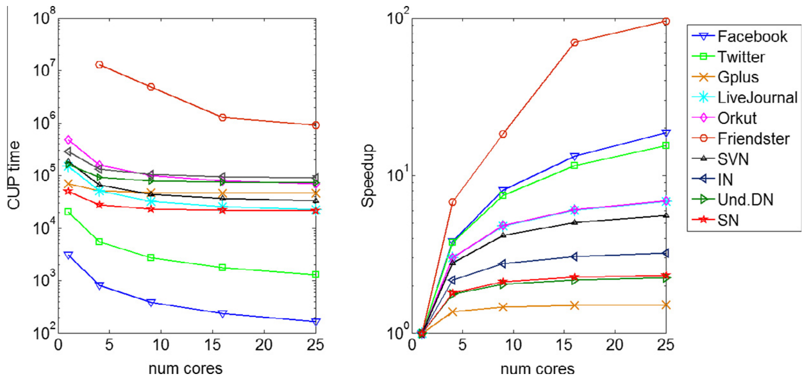

Fig. 8. CUP times and speedup for the test datasets.

dense networks (SVN and IN), FPMQA performs very well (32.7 s and 89.3 s, respectively). More community properties of these four networks will be given next. In the above examples, the number of passes is always smaller than 6.

6.2. The community structures of four networks

Fig. 9 shows the community structures of the four networks derived from our Tianya data. On the Similar-View Network, our community detection algorithm has run 5 passes. At the end of the fifth pass, we detect 1195 communities that have more than 100 users as shown in Fig. 9(a). These communities account for about 89% of all users in SVN. The modularity of this partition is 0.36, which is not very high. That is because our SVN is a dense network, the community characteristic in a dense graph is always unobvious. Our algorithm allows us to analyze more closely the only community that has a equilibrate distribution of users. As shown in Fig. 9(a), we consider the sub-communities provided by FPMQA in the fourth pass. These sub-communities are closely connected to each other and are themselves composed of heterogeneous groups of users. These groups of users might have different interests or different viewpoints to the same topics. On the Interest Network, our algorithm has also run 5 passes. We totally detect 1847 communities as shown in Fig. 9(b). The average of IN is 253.78, it is still a dense network. The modularity of the final partition is only 0.35, which means the community characteristic is not obvious in IN. While for the Und. Dense Network and the Semantic Network, the situations have changed. We detect 1617 communities with the modularity of 0.50 in the end of forth pass. The community characteristic of UDN is more obvious than those in SVN and IN. Users in the same community might communicate with each other very frequently. And in the Semantic Network, we get the highest modularity (0.62) in the end of sixth pass. The detected 1187 communities are strongly segregated with each other, as shown in Fig. 9(d).

6.3. Evaluation metric

Currently the performance of community detection methods is evaluated by manual inspection. For each detected community an effort is made to interpret it as a "real" community by identifying a common property or external attribute shared by all the members

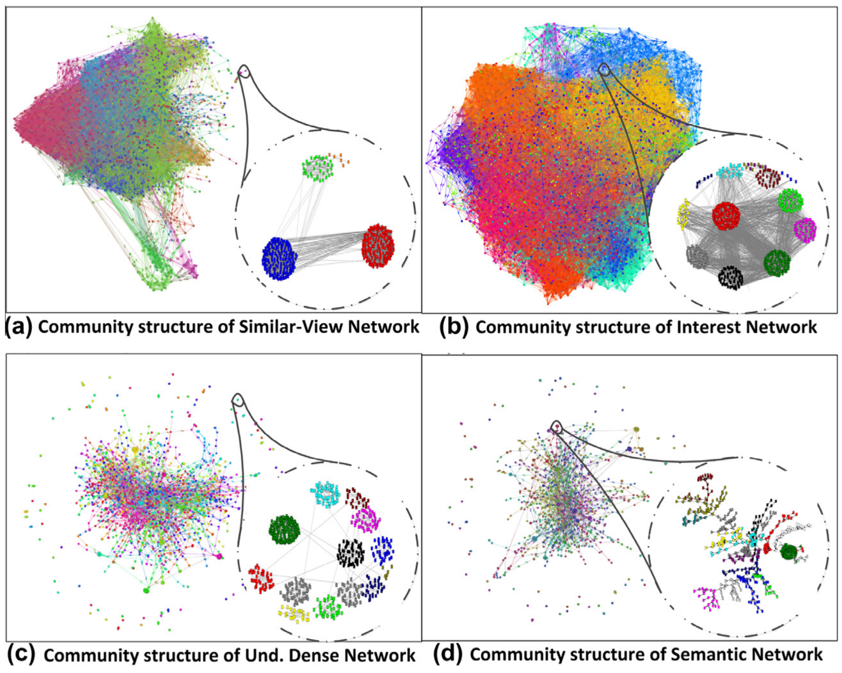
(a) Community structure of Similar-View Network
(b) Community structure of Interest Network
(c) Community structure of Und. Dense Network
(d) Community structure of Semantic Network

Fig. 9. Community structures of the four networks derived from Tianya data.

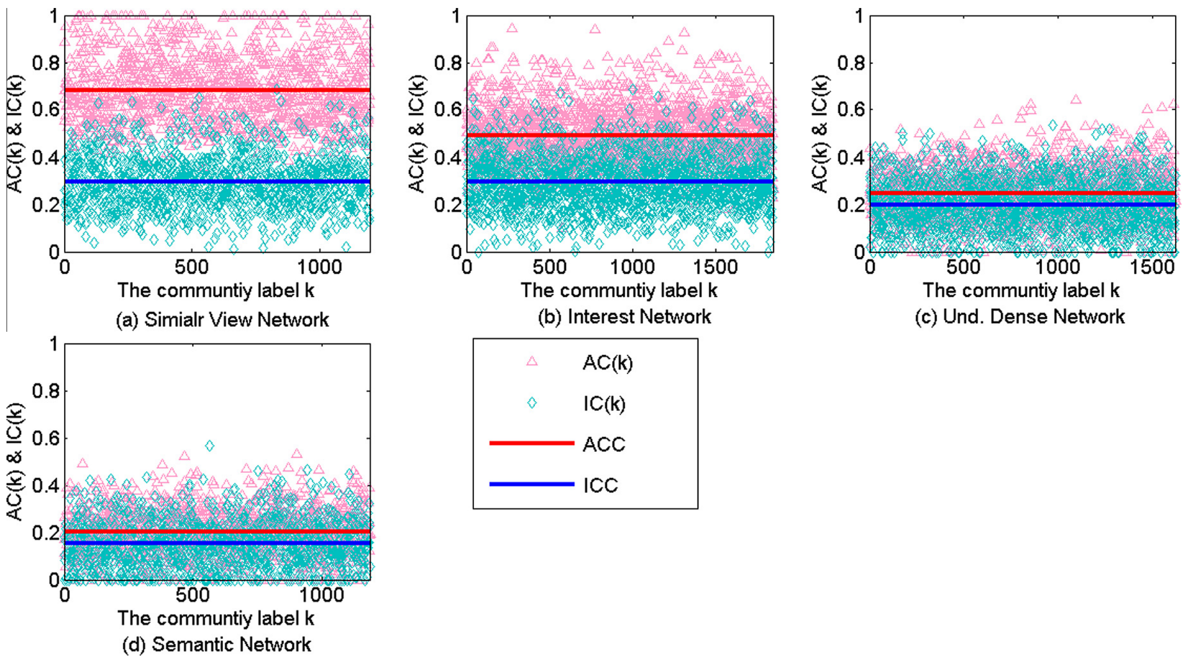

Fig. 10. The attitude consistency (ACC) and the interest consistency (ICC) of the detected communities in the four networks derived from Tianya data.

of the community. As in online social network of Tianya, we identify communities based on the structure of the network and then find that these communities correspond to groups of users with common hobbies, social functions, occupations, interests on some topics, viewpoints, etc. Some of those common properties and external attributes can be found on user profiles, which are provided by users themselves. However, not every user on Tianya is willing to provide their true and complete personal information. There exist multiple structural definitions of network communities, the lack of reliable ground truth makes it is hard to interpret and evaluate the results of community detection.

To address the above problems, we propose an evaluation metric only according to users' behaviors. In Section 3.3, we define a parameter AC_{ij} to measure the attitude consistency to together-reply topics/users of two connected users i and j. We can also use this parameter to measure the attitude consistency of community C_k, which is defined as follow:

$$AC_k = \underset{(i,j)\in C_k}{Avg}(AC_{ij}) = \frac{\sum_{(i,j)\in C_k} AC_{ij}}{m_k} \tag{11}$$

where m_k is the number of edges included in community C_k. Then the attitude consistency of the whole network is defined as

$$ACC = \underset{C_k\in C}{Avg}(AC_k) = \frac{\sum_{C_k\in C} AC_k}{\kappa} \tag{12}$$

where κ is the number of communities.

In addition, we define the interest consistency of two users i and j as follow:

$$IC_{ij} = \frac{count(T_{ij})}{Max(count(T_i), count(T_j))} \tag{13}$$

where T_{ij} is the topic set includes the topics which are together discussed by user i and j as defined in Section 2.3. T_i is the topic set includes the topics which are discussed by user i. The range of IC_{ij} is also between 0 and 1, and a higher value corresponds to a greater degree of interest consistency of the given user pair. Similarly, the interest consistency of community C_k and the whole network are defined as

$$IC_k = \underset{(i,j)\in C_k}{Avg}(IC_{ij}) = \frac{\sum_{(i,j)\in C_k} IC_{ij}}{m_k} \tag{14}$$

$$ICC = \underset{C_k\in C}{Avg}(IC_k) = \frac{\sum_{C_k\in C} IC_k}{\kappa} \tag{15}$$

A good division of our network should be thought of as a set of users that has high AC_{ij} and IC_{ij} values on edges between its members. AC_{ij} and IC_{ij} tell the interactive behaviors of users i and j. They are the ground truths, because users' behaviors are real existence. Therefore, ACC and ICC can be used as the metrics to evaluate the accuracy of the results of community detection.

Fig. 10 shows the attitude consistencies (ACCs) and the interest consistencies (ICCs) of the four networks. In the Similar-View Network, the average attitude consistency of the network is 0.68, which implies most users in the same community have similar viewpoints to their together-reply topics/users. The average interest consistency of the network is 0.29, which is not high. That is because every user on Tianya might have many hobbies, their interests may not completely overlap. The same evolution manipulation is deduced to the Interest Network, Und. Dense Network and the Semantic Network. The average attitude consistency of the Interest Network is 0.49, which is significantly below that of the Similar-View Network. That means even two persons have similar interests, they do not necessarily have to become good friends. ACCs of the Und. Dense Network and the Semantic Network are 0.25 and 0.20 separately, the values of which are very low. That is because even two IDs communicate very frequently and friendly based on a given topic, they do not necessarily have to become good friends either. The average interest consistencies of the three networks are 0.29, 020 and 0.15. As expect, those networks are built according to the actual interactions of users. It is difficult to centralize users with completely overlapping interests together only according to the network topology. Therefore, the interest consistency of the network is high if ICC researches 0.2. Our method based on the Similar-View Network is better than other models in finding communities in online social network.

7. Summary and conclusion

The overall detection approach is summarized as Algorithm 2.

Algorithm 2. Online community detection based on Similar-View Network model and FPMQA

```
Input: Social network data set (One theme discussion or
    More),
Output: Online communities identified
1: Construct the Interest Network using given social network
    data set;
2: Mine the semantic information and update the Interest
    Network using the attitude consistence value;
3: Call the fast parallel modularity optimization algorithm
    (FPMQA);
4: Return: Online communities identified
```

Compared to traditional methods, our approach has three advantages: (1) it conforms to the practical meanings of online community; (2) by taking the parallel strategy, it reduces the running time of community detection; and (3) using the reliable ground truths to evaluate detected communities, which proves the high effective of our method.

As shown in Fig. 11, we assume that the real groups have been known and labeled in Fig. 11(a). That is, IDs A, C, E and F are from group G1, IDs B, D and G are from group G2. We notice in Fig. 11(a) that, IDs from the same group often reply to the same topic or comment. For instance, IDs B, D and G together comment to ID A twice. To grasp this property, we build the Interest Network as shown in Fig. 11(b). Then, we detect the attitude consistence of two connected IDs in IN, and update original network. Fig. 11(c) shows the Similar-View Network in which connected IDs have the same viewpoints to their together-reply topics/users. For example, IDs B, D and G's attitudes are same to ID A. Finally, we apply the fast parallel modularity optimization algorithm, and get communities we expect as shown in Fig. 11(d). IDs belonging to the same community have properties in common: similar interests, consistent viewpoint to certain topics/users. It conforms to the practical meanings of online community.

Another advantage of our approach is the running time of community detection task. It combines the advantages of CNM [7] and FUC's [6] and has several unique features. CNM finds the pair of communities with the "global" maximum ΔQ, however, finding this "global" maximum ΔQ is time consuming. Our method select the "local" maximum ΔQ as FUC does, but the definition of the "local area" of every community is more reasonable. FUC only considers the community pair between C_i and its neighboring communities. That may bring the final modularity declination in some cases, as stated in Section 5.3. What's more, our algorithm takes the parallel strategy, it has a running time of $O(k^{max}(k^{max} + \langle k\rangle > \log k^{max}))$. It can be applied to the community detection task in a large scale dense network, if the maximum degree of the network is not very high.

Finally, we propose two reliable metrics to evaluate the detected communities, which are the average attitude consistency *ACC* and the average interest consistency *ICC*. We do not need to spend extensive manual effort to distinguish the real true information from others in users' profiles. We only need to

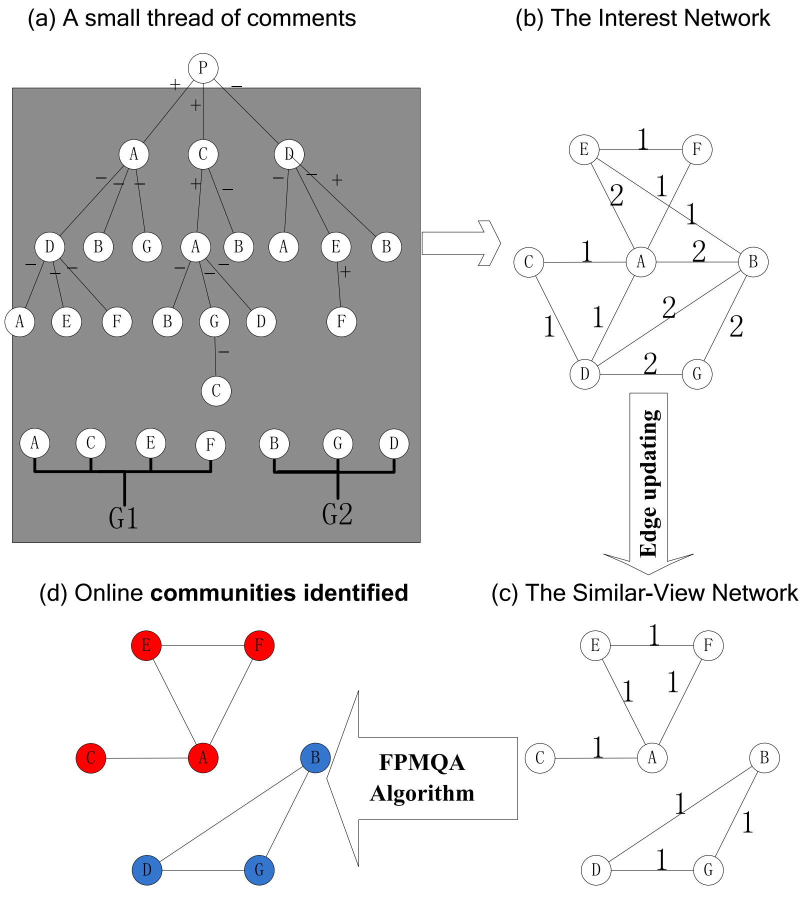
(a) A small thread of comments
(b) The Interest Network
(c) The Similar-View Network
(d) Online communities identified
Edge updating
FPMQA Algorithm
G1
G2

Fig. 11. An example to illustrate the advantages of our approach.

ARTICLE IN PRESS

consider interactive behaviors of users *i* and *j*, which is AC_{ij} and IC_{ij}. Those ground truths are real existence, and prove that our method based on the Similar-View Network is better than other models in finding communities in online social network.

Acknowledgments

This work was supported by Funding of Jiangsu Innovation Program for Graduate Education (Project No: CXZZ12_0162), the Fundamental Research Funds for the Central Universities.

References

[1] Z. Bu, Z. Xia, J. Wang, A sock puppet detection algorithm on virtual spaces, Knowledge Based Systems 37 (2013) 366–377.

[2] Z. Bu, Z. Xia, J. Wang, C. Zhang, A last updating evolution model for online social networks, Physica A: Statistical Mechanics and Its Applications 392 (9) (2013) 2240–2247.

[3] Z. Bu, C. Zhang, Z. Xia, J. Wang, An FAR-SW based approach for webpage information extraction, Information Systems Frontiers, SI: Information Reuse, Integration, and Reusable Systems (2013) (published online).

[4] M. Boguna, R. Pastor-Satorras, Epidemic spreading in correlated complex networks, Physical Review E 66 (2002) 047104.

[5] M. Boguna, R. Pastor-Satorras, A. Vespignani, Absence of epidemic threshold in scale-free networks with degree correlations, Physical Review Letter 90 (2003) 028701.

[6] V.D. Blondel, J.L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, Journal of Statistical Mechanics: Theory and Experiment (2008) P10008.

[7] A. Clauset, M.E.J. Newman, C. Moore, Finding and evaluating community structure in networks, Physical Review E 69 (2004) 026113.

[8] J. Duch, A. Arenas, Community detection in complex networks using extremal optimization, Physical Review E 72 (2005) 027104.

[9] L. Danon, J. Duch, A. Diaz-Gulera, A. Arenas, Comparing community structure identification, Journal of Statistical Mechanics: Theory and Experiment (2005) P09008.

[10] R. Guimerà, L. Danon, A. Díaz-Guilera, F. Giralt, A. Arenas, Self-similar community structure in a network of human interactions, Physical Review E 68 (2003) 065103.

[11] P. Gleiser, L. Danon, Community structure in jazz, Advances in Complex Systems 6 (4) (2003) 565–573.

[12] M.E.J. Newman, Detecting community structure in networks, The European Physical Journal B – Condensed Matter and Complex Systems 38 (2) (2004) 321–330.

[13] V. Gomez, A. Kaltenbrunner, V. Lopez, Satistical analysis of the social network and discussion threads in Slashdot, in: Proc. of the 17th Intl. Conf. on World Wide Web, 2008, pp. 645–654, doi: http://dx.doi.org/10.1145/1367497.1367585.

[14] M. Girvan, M.E.J. Newman, Community structure in social and biological networks, Proceedings of the National Academy of Sciences of the United States of America 99 (12) (2002) 7821–7826.

[15] R. Kanai, B. Bahrami, R. Roylance, G. Rees, Online social network size is reflected in human brain structure, Proceedings of the Royal Society B 279 (2012) 1327–1334.

[16] Z. Kou, C. Zhang, Reply networks on a bulletin board system, Physical Review E 67 (2003) 036117.

[17] K. Lewis, M. Gonzalez, J. Kaufman, Social selection and peer influence in an online social network, Proceedings of the National Academy of Sciences of the United States of America 109 (1) (2012) 68–72.

[18] J. McAuley, J. Leskovec, Learning to Discover Social Circles in Ego Networks, Neural Information Processing Systems, 2012.

[19] A. McCallum, A. Corrada-Emmanuel, X. Wang, Topic and role discovery in social networks, in: Proc. of the 19th Intl. Conf. on, Artificial Intelligence, 2005, pp. 786–791.

[20] M.E.J. Newman, Assortative mixing in networks, Physical Review E 89 (2002) 208701.

[21] M.E.J. Newman, Mixing patterns in networks, Physical Review E 67 (2003) 026126.

[22] P. Pons, M. Latapy, Computing communities in large networks using random walks, Journal of Graph Algorithms and Applications 10 (2) (2006) 191–218.

[23] U.N. Raghavan, R. Albert, S. Kumar, Near linear time algorithm to detect community structures in large-scale networks, Physical Review Letter 76 (2007) 036106.

[24] S.N. Soffer, A. Vazquez, Network clustering coefficient without degree-correlation biases, Physical Review E 71 (2005) 057101.

[25] T. Tian, R. Hankins, J. Patel, Efficient aggregation for graph summarization, in: Proc. of the 2008 ACM SIGMOD Intl. Conf. on Management of Data, 2008, pp. 567–580.

[26] Z. Xia, Z. Bu, Community detection based on a semantic network, Knowledge Based Systems 26 (2012) 30–39.

[27] J. Yang, J. Leskovec, Defining and evaluating network communities based on ground-truth, in: Proc. of 2012 IEEE Intl. Conf. on Data Mining, 2012, doi: http://dx.doi.org/10.1145/2350190.2350193.

[28] J. Zeng, S. Zhang, C. Wu, A framework for WWW user activity analysis based on user interest, Knowledge-Based Systems 27 (8) (2008) 905–910.

[29] Z. Zhao, S. Feng, Q. Wang, J.Z. Huang, G.J. Williams, J. Fan, Topic oriented community detection through social objects and link analysis in social networks, Knowledge-Based Systems 26 (2012) 164–173.

[30] W.W. Zachary, An information flow model for conflict and fission in small groups, Journal of Anthropological Research 33 (4) (1977) 452–473.