Information Sciences 432 (2018) 164–184

Contents lists available at ScienceDirect

Information Sciences

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

Overlapping community detection in heterogeneous social networks via the user model

Mingqing Huang[a], Guobing Zou[a,*], Bofeng Zhang[a,*], Yue Liu[a], Yajun Gu[a], Keyuan Jiang[b]

[a] School of Computer Engineering and Science, Shanghai University, Shanghai, 200444, China
[b] Department of Computer Information Technology and Graphics, Purdue University Northwest, Indiana, 46323, USA

ARTICLE INFO

Article history:
Received 22 April 2016
Revised 8 November 2017
Accepted 28 November 2017
Available online 2 December 2017

Keywords:
Community detection
Heterogeneous network
User modeling
Community seed
Community expansion

ABSTRACT

Clustering users with more common interests who interact frequently on social networking sites has attracted much attention from researchers due to the high economic value and further application prospects. Community detection is a widely accepted means of dealing with the challenge of clustering users, but conventional methods are inadequate since there are billions of vertices and various relations in social media. Through the user model, a heterogeneous network containing both undirected and directed edges is built in this study to exactly simulate a social network. A novel approach for overlapping community detection in a heterogeneous social network (OCD-HSN) is proposed, which contains seed selecting and community initializing and expanding to accurately and efficiently unfold modules in parallel. Experimental results on artificial and real-world social networks demonstrate the higher accuracy and lower time consumption of the proposed scheme compared with other existing state-of-the-art algorithms.

1. Introduction

With the rapid development of Web 2.0 technology, the widespread acceptance and utilization of social networking sites (SNSs) have changed the modes of communication by facilitating the ability of individuals from diverse areas to interact and share interests. Social networks such as Twitter, Facebook, Google+, Orkut, LinkedIn, Friendster and Multiply have attracted billions of users, and have also received attention from many researchers and media. Whereas these sites offer potential benefits to users [22], they also pose a challenging problem of information overload and place higher requirements on community detection.

Communities, also known as clusters or modules, are the sets of vertices with common properties or the same roles in the network [12]. Community detection is a process of identifying all communities in a network. A large variety of techniques is available to carry out this single task. For more information, see the recent survey by Schaffer [33]. The ability to unfold these sub-structures in a social network can provide insight into how the network topology and functions affect each other, and can further provide theoretical support for interest prediction, friend recommendation, and event evolution.

* Corresponding authors.
E-mail addresses: mqhuang@shu.edu.cn (M. Huang), gbzou@shu.edu.cn (G. Zou), bfzhang@shu.edu.cn (B. Zhang), yliu@staff.shu.edu.cn (Y. Liu), yjgu@shu.edu.cn (Y. Gu), kjiang@pnw.edu (K. Jiang).

https://doi.org/10.1016/j.ins.2017.11.055
0020-0255/

Entities in social systems are treated as vertices, and edges are connected between vertices based on various relations. With such a representation, the theories of complex networks can be applied to solve many specific problems; this has currently become the leading method in pertinent research efforts [13]. How to construct appropriate complex networks based on the characteristics of social media and explore effective community detection algorithms in large-scale social networks, however, has become challenging. This study aims at meeting this challenge.

The user model, also called the user profile, as a mirror of a real user in the cyber-world, is constructed to depict basic user attributes, interests, and social relations. Although the *Vector Space Model* [19] has been widely adopted to model users, it suffers from several critical weaknesses. It models users in a bloated form, has no uniform representation standard, and cannot implement semantic expansion. To address these shortcomings, the semantic ontology representation [9] is selected in this study to model users' semantic interests, and social relations among users are expressed independently of users' interests.

The Pearson correlation coefficient is applied to calculate the interest similarity value between user models on the corresponding layer of the semantic ontology, and the interest similarity values of multiple layers are mixed using the hyperbolic tangent function. We combine interest similarities and social interactions in the same network to construct a heterogeneous network that simultaneously contains undirected and directed edges.

In this study, it is demonstrated that the community detection in heterogeneous networks using seed set expansion can not only help mine overlapping and hierarchical communities, but can also calculate membership degrees of overlapping vertices in different communities. To the best of our knowledge, this is the first attempt to study overlapping community detection in heterogeneous social networks (OCD-HSN) containing both undirected and directed edges.

The main characteristics and innovations of this study are the following:

- The interest similarities and social interactions among user models are both included in the same heterogeneous network.
- A novel approach is proposed to detect communities in heterogeneous networks containing both undirected and directed edges.
- Multiple seeds are selected in the heterogeneous network to facilitate community detection in parallel.
- Each community is initialized to a vertex set according to the seed vertex characteristic to improve community accuracy and accelerate detection speed.
- The election principle is applied to add and remove multiple vertices during iterative expanding of communities, to further accelerate community detection.

To assess the accuracy and effectiveness of our method, we have compared it with two other existing state-of-the-art algorithms. Experimental results on both artificial and real-world social networks indicate that our proposed approach outperforms the other two methods with higher accuracy and lower time complexity. With these advantages, our approach can well satisfy the application requirements of large-scale social networks with billions of vertices.

The rest of this paper is organized as follows. In Section 2, we summarize related work on the techniques of user modeling and community detection. We introduce the user modeling method and build a heterogeneous social network in Section 3. In Section 4, the proposed community detection approach is described in depth. The experimental results of the proposed scheme are compared and analyzed in Section 5. Finally, we conclude the paper in Section 6 by discussing plans for future work.

2. Related work

Different from traditional complex networks, social networks express users of SNSs and the relations among them. By analyzing such networks, not only is it possible to gain insights into social phenomena and processes that take place in the real world, but one can also extract actionable knowledge that can be beneficial in retrieval tasks and information management, such as online content navigation and recommendation systems.

2.1. User model

A user model is a visual representation of personal data associated with a specific user, and is broadly adopted in the domain of personalization services.

Many methods based on the user model cannot accurately describe user attributes. The approach of recommending microblogs based on the user model [27] does not provide insights into the social relations among user models. The research on theme recommendation in microblogging scenarios based on neighborhood-user profiles [37] merely binds the semantic interest expansions and the social relations of a user model. Twitter user profiling [18] adopts the community information only when the group demonstrates distinct characteristics, and detects clusters in the multi-hop network of the target user instead of the entire network, leading to loss of exact community information.

2.2. Community detection in complex networks

The Girvan–Newman method [14] identifies edges lying between communities and then removes them, thereby returning results of reasonable quality, but it runs very slowly. The method of community detection in directed networks using

spectral clustering [23] is unable to unfold overlapping communities. By dealing with the set of maximal cliques and adopting an agglomerative rule, the EAGLE algorithm [34] detects overlapping and hierarchical properties of community structure together, but it is incapable of being extended to directed networks and can only be applied to highly dense networks. The community detection approach using the genetic algorithm [29] allows for the detection of overlapping communities and the calculation of membership degrees of overlapping vertices, but it cannot yet detect hierarchical structures. The method of *Local Fitness Maximization* [21] and the integrative approach to determine modules [20] can extract overlapping and hierarchical structural units in networks, but both ignore the belonging factors of overlapping vertices in different communities. The local community method via PageRank random walk [15] concurrently starts from selected seeds and rapidly extracts communities in large-scale networks, but the community results are rather inconsistent with the factual module structures.

In short, none of the above-mentioned methods are capable of extracting communities in a heterogeneous network containing both undirected and directed edges.

2.3. Community detection in social networks

Recently, increasing attention has been focused on community detection in social networks, not only as a means of uncovering the underlying phenomena taking place in such systems, but also to allow exploitation of its principles in a wide range of intelligent services and applications, such as automatic event detection in social media content [30].

Based on the optimization of modularity, an efficient approach [8] has been proposed to unfold hierarchical communities that adapts to networks of unprecedented size. However, it suffers from the issue of resolution limitation. The multilevel hierarchical kernel spectral clustering method [25] automatically obtains quality modules at different levels of granularity in large-scale real-life networks. The ongoing problem to be solved is that it is incapable of revealing the overlapping properties. Overlapping community detection in social networks [16] brings the user model into community extraction, but the social interactions among users are converted into similar interest relations, requiring further improvements to accuracy and efficiency.

Although most of these methods detect communities in social networks, they have not deeply taken user features into account, and have failed to execute appropriate research on SNSs and Web Mining.

3. Heterogeneous social networks

In this section, we generate a heterogeneous social network to provide the foundation for community detection. First, each user is exhibited as a user model with semantic interests and social interactions by analyzing the features of users. We then calculate the interest similarities and connect user models based on the social relations among them, yielding a graph with vertices representing user models and edges representing various relations.

3.1. User modeling

We select Chinese microblogging as the research platform in this study. However, the proposed approach with a slight modification can be applied to any microblogging and even to social media platforms programmed for other languages.

3.1.1. Ontology base

A general ontology instead of the domain one is adopted since the user behavior records on a microblogging platform are not dependent on a single field, but involve all aspects of real life.

The *Baidu Encyclopedia* [1] is the most comprehensive and influential Chinese ontology, as it was developed in an attempt to create a Chinese information collection platform covering all fields of knowledge. In the January 2015 edition of the *Baidu Encyclopedia* ontology base, there are 11 keyword channel classifications in the first layer, 89 in the second layer, and 380,627 keywords in all four layers, as shown in Fig. 1. Because of these characteristics, the *Baidu Encyclopedia* is referred to as the ontological fundamental for user modeling in this study.

3.1.2. User interest quantization

Based on the microblogging histories of users, we quantify user interests by interest degrees on specific keywords in the ontology base.

Term frequency - inverse document frequency (*TF-IDF*) [31] is adopted to construct user interest models. *TF-IDF* is illustrated as

$$TF\text{-}IDF(D_i, t_j, D) = \frac{tf(t_j, D_i)}{\sum_{t_k \in D_i} tf(t_k, D_i)} \times idf(t_j, D). \tag{1}$$

Where $TF\text{-}IDF(D_i, t_j, D)$ represents the *TF-IDF* value of keyword t_j to document D_i in document collection D; $tf(t_j, D_i)$ denotes the number of times that term t_j appears in document D_i; and $idf(t_j, D)$ is given by

$$idf(t_j, D) = log\left(\frac{|D|}{|\{D_k \in D : t_j \in D_k\}|}\right). \tag{2}$$

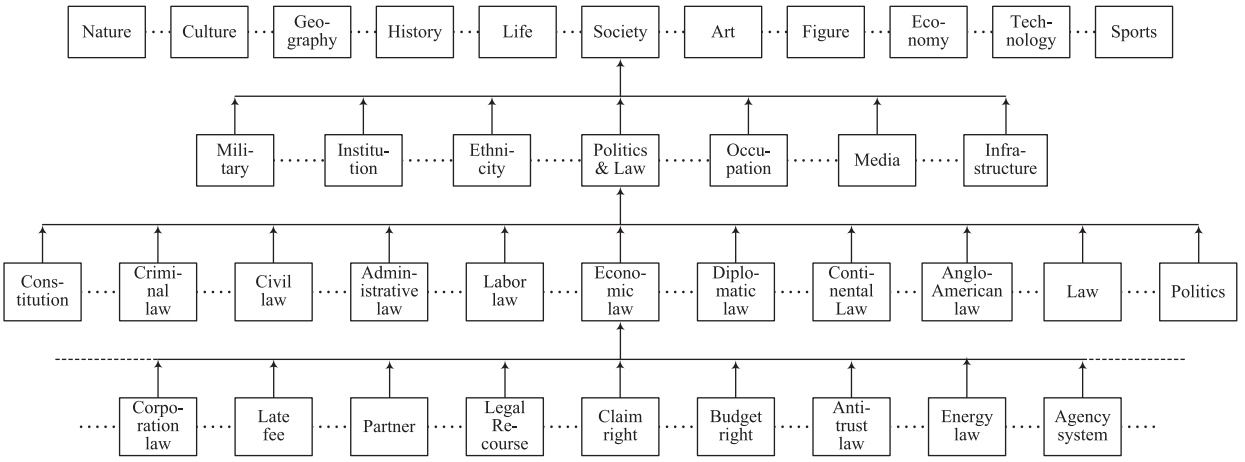


Fig. 1. User model ontology base with sibling keywords connected to each other with "⋯" and child keywords pointing at their parents with "↑".

The process of quantifying the degrees of user interests consists of the following four steps.

Step 1: Interest mining

The recent (three months, for example) microblogs of each user are accumulated and stored in a separate document for word segmentation. Then, those keywords that can be searched from our constructed ontology base are extracted, and those not included in the ontology base are discarded.

Step 2: Degree quantization

The *TF-IDF* value of each keyword in each document is calculated, and all *TF-IDF* values associated with the same user are transferred to the corresponding user interest model's four-layer ontology base, which we called the initial user interest model of this user.

Step 3: Value propagation

The direct use of the initial user interest model does not reflect the semantic relations among keywords, yet requires heavy computation. Therefore, the final user interest model in this study retains only the top two-layer ontology base, with up to 100 keywords. This is not to say that the bottom two-layer ontology base is removed, but that the interest degrees on keywords of the bottom two-layer are propagated to the ones of the top two-layer.

The *One-way Upward Diffusion Model* is adopted to propagate interest degrees from the lower layers to the upper layers. That is, the interest degree on a keyword t_i in the fourth layer is multiplied by a factor α_i and added to the one on its parent keyword, and so forth. Thus, the propagation can then be done in the third and second layers, where α_i is the reciprocal of the number of sibling keywords of keyword t_i. The diffusion model is shown as

$$I'_j = I_j + \sum_{i=1}^{n^j_{child}} \left(\alpha_i I^j_i\right) = I_j + \sum_{i=1}^{n^j_{child}} \left(\frac{I^j_i}{n^j_{child}}\right). \tag{3}$$

Where I_j is the initial interest degree on keyword t_j; I'_j represents the final one after the propagation is done; n^j_{child} denotes the number of child keywords; and I^j_i indicates the interest degree on the i-th child keyword.

Step 4: Model reduction

After the propagation has been done, only the top two-layer keywords and the relevant interest degrees are retained for representation of user interests.

Without loss of generality, the sum of interest degrees in each layer is then normalized; that is, the sum is equal to 1 after uniform scale zooming.

3.1.3. User social attribute tagging

The more times and closer to the current time the forwarding and commenting of posts are, the more intimate the relations between relevant users are. These frequently occurring interactions are built based on social relations of follower-followee, so the interactions among users are asymmetric. For simplicity, in this study the social attributes of the user model contain followers and followees only. In particular applications, such as collaborative filtering systems, which need to include the affinity degrees of social relations, see the related literature [3].

3.1.4. User model representation

After user interest quantifying and social relation tagging, our formal user model consists of three correlated parts:

1) Basic attributes of the user, such as account ID, nickname, gender, registered address, labels, etc.
2) Interest degrees on 100 keywords of the *Baidu Encyclopedia* top two-layer ontology base.
3) Social attributes that contain followers and followees of the user.

3.2. Heterogeneous social network construction

Given user models, their interest similarities among users can be calculated and represented with undirected edges. In addition, social interactions can be expressed with directed edges. As a result, we build the heterogeneous social network with both undirected and directed edges by treating user models as vertices.

3.2.1. Interest similarity match-making

For match-making among users based on their interests, we first calculate the interest similarity in the corresponding ontology layer, and then mix the interest similarities in multiple layers. Finally, the similar interest relations are converted into the edges among user models by setting a well-chosen threshold. Specific steps are as follows.

Step 1: Interest similarity calculation

The Pearson correlation coefficient is employed to calculate the interest similarity value in the corresponding layer of semantic ontology between every two user models; the formal representation of the expanded version for our study is described as follows:

$$I(u_p, h, i) = u_p(h, i) - \bar{u}_p(h). \tag{4}$$

$$S(u_p, u_q, h) = \frac{\sum_{i=1}^{t(h)} I(u_p, h, i) I(u_q, h, i)}{\sqrt{\sum_{i=1}^{t(h)} I^2(u_p, h, i)} \sqrt{\sum_{i=1}^{t(h)} I^2(u_q, h, i)}}. \tag{5}$$

Where $u_p(h, i)$ denotes the interest degree of user model u_p on the i-th keyword of the h-th ontology layer; $\bar{u}_p(h)$ expresses the average interest degree of u_p on all keywords of the h-th ontology layer; $S(u_p, u_q, h)$ represents the interest similarity value between the user model u_p and u_q in the h-th ontology layer; and $t(h)$ is the number of keywords in the h-th ontology layer.

Step 2: Interest similarity mixing

The final user model contains keywords of the top two-layer ontology base, and apparently the interest similarity of the second layer is more important than that of the first layer. The hyperbolic tangent function is adopted to address this imbalance, which is defined as

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \tag{6}$$

The value domain of the hyperbolic tangent function is $(-1, 1)$, and the differential coefficient is smaller and smaller within the definition domain $[0, +\infty)$, as depicted in Fig. 2. This is consistent with the fact that the probabilities of keywords of two adjacent layers appearing in practical applications converge to an upper bound with the increase of the sequence number of the *Baidu Encyclopedia* ontology layer.

As shown in Eqs. (7)–(9), the mean value $M(\varphi)$ of all the sequence numbers of ontology layers is calculated first, and then the influence powers of the interest similarity values of different ontology layers on the final one are adjusted by a parameter γ.

$$M(\varphi) = \frac{\sum_{h=1}^{\varphi} h}{\varphi}. \tag{7}$$

$$W(h) = \frac{e^{h\gamma^{[h-M(\varphi)]}} - e^{-h\gamma^{[h-M(\varphi)]}}}{e^{h\gamma^{[h-M(\varphi)]}} + e^{-h\gamma^{[h-M(\varphi)]}}}. \tag{8}$$

$$S(u_p, u_q) = \frac{\sum_{h=1}^{\varphi} W(h) S(u_p, u_q, h)}{\sum_{h=1}^{\varphi} W(h)}. \tag{9}$$

Where φ denotes the number of ontology layers and $W(h)$ represents the weight value of the interest similarity of the h-th ontology layer. γ is determined by the ratio of the number of keywords in the highest layer and that in the lowest layer. The *Baidu Encyclopedia* includes 89 and 11 keywords in the second and first layer separately, so we set $\gamma = 64$ to derive $W(2)/W(1) \approx 89/11$. $S(u_p, u_q)$ indicates the final interest similarity value between the user model u_p and u_q.

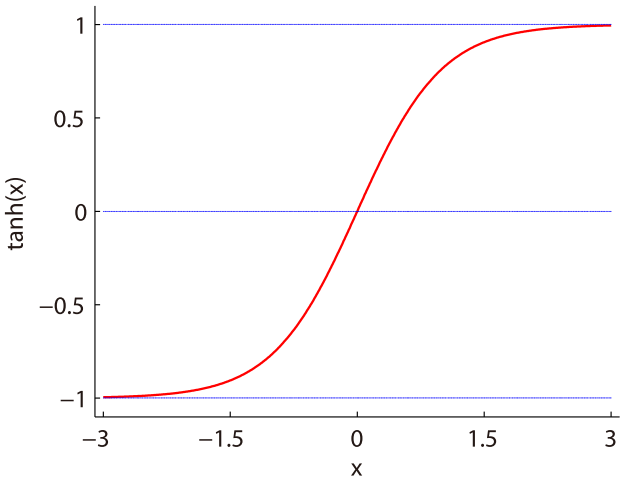


Fig. 2. Progressive trend of hyperbolic tangent function.

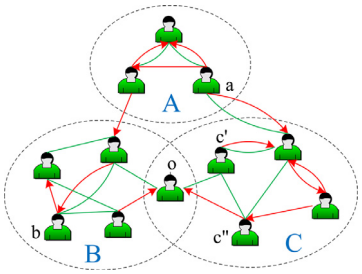


Fig. 3. Constructed heterogeneous social network with different communities represented by dashed ovals.

Step 3: Interest edge connection

When the interest similarity value is low enough, it demonstrates little meaningfulness in realistic applications. An appropriate threshold is selected using machine learning based on the requirements of a specific application. We then connect an unweighted and undirected edge between user models if the related interest similarity value is greater than the threshold. Otherwise we ignore the related similar interest relation.

3.2.2. Social interaction designating

Not involving in any specific applications, in this study we ignore the forwarding, commenting and relevant frequencies in recent periods, and only consider the social relations of follower-followee between user models.

Directed edges are exploited to denote the social interaction relations with arrows pointing from followers to followees.

3.2.3. Heterogeneous social network connection

As pictured in Fig. 3, the user interest network developed in Section 3.2.1 is mingled with the user interaction network mentioned in Section 3.2.2 to generate a heterogeneous social network based on the uniqueness of user account ID. Not only do similar interest relations exist, expressed by undirected edges, but social interaction relations represented by directed edges also exist in the same network.

Since the heterogeneous social network is especially relevant to practical applications, undirected and directed edges are multiplied by weight values w^{undir} and w^{dir} separately to cater to the different needs of specific applications. The optimal combination of w^{undir} and w^{dir} can be found through machine learning in practical applications. Since this study does not involve particular applications, for simplicity's sake we set $w^{undir} = 1$ and $w^{dir} = 1$.

4. Community detection approach

In this section, we first present the framework of our proposed approach. Next, we describe in detail the multiple seeds selection strategy, overlapping community detection algorithm, and membership degree calculation method in the heterogeneous social network. Finally, we reveal how to discover the hierarchical module structure and analyze the time complexity.

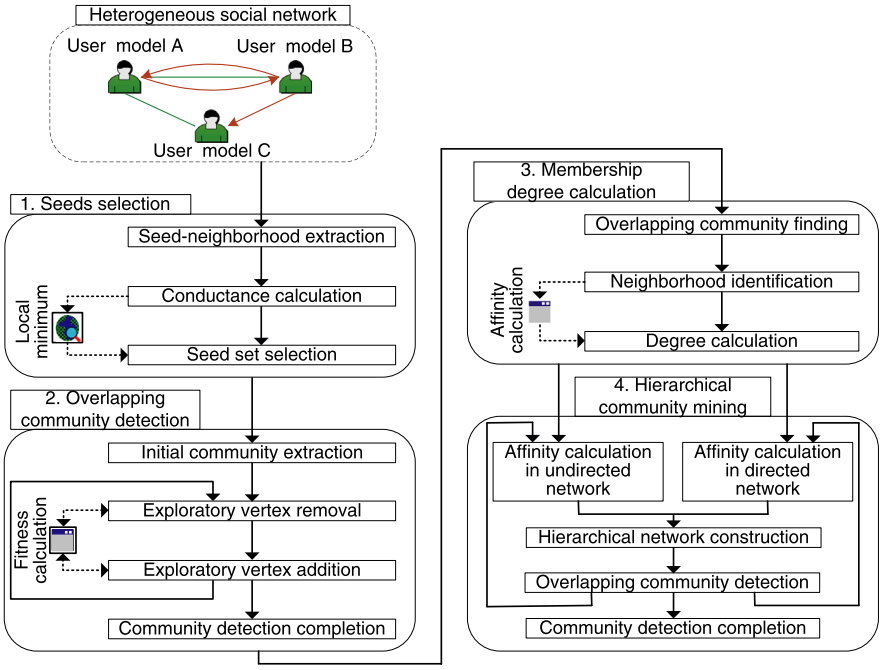


Fig. 4. Overall framework of our approach for community detection.

4.1. Methodological framework

The detailed overall framework of community detection is illustrated in Fig. 4, where the construction of the heterogeneous social network has been plugged in as the foundation. First, the seed vertex set is selected based on local minimum conductance, to carry out parallel computation on multi-core computers. Then, we detect overlapping communities using initial community set expansion and optimization [35], which depends on the local fitness maximization via multiple-vertex removal and addition. After that, the membership degrees overlapping vertices belonging to different communities are calculated through the affinity with neighborhood. Finally, in terms of granularity, the discovered communities are further treated as vertices, such that the relations among different clusters are converted into edges for hierarchical community detection.

4.2. Multiple seeds selection

According to Newman's modularity principle [28], each community has a core with the other vertices as subsidiaries of the core, which plays a leading role in the community. We select most vertices of each core as the initial community, and then adopt the method of *Exploratory Vertex Removal and Addition* (*EVRA*) to identify the corresponding community. Here, we give two correlative definitions.

Definition 1 (Seed-neighborhood)**.** The seed-neighborhood of vertex v is defined as a vertex set that contains the adjacent ones connected by undirected edges, the followee ones and v itself. v is regarded as the central vertex of this seed-neighborhood.

Definition 2 (Conductance)**.** Extending the concept of conductance in unweighted and undirected networks, we define the conductance $\phi(S)$ in a heterogeneous network as

$$cut(S) = cut_{dir}(S) + cut_{undir}(S). \tag{10}$$

$$vol(S) = vol_{dir}(S) + vol_{undir}(S). \tag{11}$$

$$\phi(S) = \frac{cut(S)}{min(vol(S), vol(\bar{S}))}. \tag{12}$$

Where S denotes a connectivity vertex set in the heterogeneous network; $cut_{dir}(S)$ is twice the number of directed edges pointing from vertices in S to other external ones; $cut_{undir}(S)$ contains the number of undirected edges connecting vertices in S to other external ones; $vol_{dir}(S)$ indicates twice the sum of out-degrees of vertices in S attaching to directed edges;

$vol_{undir}(S)$ expresses the sum of degrees of vertices in S attaching to undirected edges; and $\bar{S}$ represents the complement set of S, composed of vertices out of S. In large-scale networks, since $vol(S) << vol(\bar{S})$, we just need to calculate $vol(S)$.

Based on the peak principle [20], we select the central vertex that does not belong to any detected community, of which the conductance of the seed-neighborhood is a conditional local minimum, as a seed. The conditional local minimum conductance is determined by

$$\phi(N_1(v)) \leq \phi(N_1(w)). \tag{13}$$

Where $N_1(v)$ denotes the seed-neighborhood of vertex v, and w is every vertex except v in $N_1(v)$ which does not belong to any detected community.

As presented in Fig. 3, the user models a, b, c' and c'' are selected as seed vertices. With a or b as the central vertex, the conductance of the seed-neighborhood is smaller than that with the any other vertex as the central vertex; but when we switch to c' or c'', the conductance is not greater only.

4.3. Overlapping community detection

4.3.1. Integrative approach

Based on the identification of central vertices as multiple seeds, the corresponding seed-neighborhoods are generated as initial communities. Then, we adopt the *EVRA* method to detect overlapping communities.

Definition 3 (Neighborhood)**.** The neighborhood of community C is defined as a vertex set that contains all those connecting to vertices in C by undirected or directed edges but not belonging to C.

In our heterogeneous social network, we calculate the fitness of community based on the idea in [21], which can be used for affirming a community.

Definition 4 (Community fitness)**.** The fitness f_C of community C is calculated to express the cohesion degree by a community fitness function, defined as

$$W_{out}^{C} = W_{out_{undir}}^{C} + W_{out_{dir}}^{C}. \tag{14}$$

$$f_C = \frac{W_{in}^{C}}{W_{in}^{C} + W_{out}^{C}}. \tag{15}$$

Where $W_{out_{undir}}^{C}$ is equal to half the number of undirected edges connecting vertices in C to other external ones; $W_{out_{dir}}^{C}$ represents the number of directed edges pointing from vertices in C to other external ones; and W_{in}^{C} denotes the number of undirected and directed edges in community C.

$f_C^p = f_{C+\{p\}} - f_{C-\{p\}}$ indicates the contribution of vertex p to the cohesion degree of community C. $C+\{p\}$ denotes that C includes p, and $C-\{p\}$ describes the opposite situation that p is excluded from C.

The concrete steps of our integrative approach to detect an overlapping community are presented in Algorithm 1.

Algorithm 1: Overlapping-community-detection(v).

Input: seed vertex v;
Output: community vertex set $S(C)$;

community vertex set $S(C) \leftarrow$ seed-neighborhood of vertex v;
Loop: $S(C) \leftarrow S'(C)$ of *Exploratory-removal*($S(C)$, v, τ);
Boolean *expandable* $\leftarrow$ *expandable*$''(C)$, $S(C) \leftarrow S''(C)$ of *Exploratory-addition*($S(C)$, σ);
if *expandable is TRUE* **then**
    goto **Loop**;
else
    return $S(C)$;

Functions *Exploratory-removal*($S(C)$, v, τ) and *Exploratory-addition*($S(C)$, σ), denoting the vertex removal and addition processes separately, are then explicitly demonstrated in the next two sub-subsections.

4.3.2. Exploratory vertex removal

Since there may be impurity vertices included in the community, we prune them in *Exploratory-removal*($S(C)$, v, τ) as illustrated in Algorithm 2, which is employed to perform the task of *Exploratory Vertex Removal* in our integrative approach.

The removal coefficient τ is analyzed in detail in Section 4.3.5.

In the process of *Exploratory Vertex Removal*, we no longer check whether or not to remove v, since the *EVRA* approach aims to mine the community that the seed vertex v belongs to.

Algorithm 2: Exploratory-removal($S(C)$, v, τ).

Input: community vertex set $S(C)$, seed vertex v, removal coefficient τ;
Output: community vertex set $S'(C)$;

Loop: integer $threshold \leftarrow 1$;
if $\tau|S(C)| > 1$ **then**
  $threshold \leftarrow \lfloor \tau|S(C)| \rfloor$;
foreach $v' \in S(C) \wedge v' \neq v$ **do**
  divide $S(C)$ into v' and the vertex set $S(C - \{v'\})$;
  if $f_{C-\{v'\}} > f_C$ **then**
    add v' to the potentially removable vertex set S_r;
if $|S_r| > threshold$ **then**
  remove $threshold$ number of vertices from $S(C)$ that are contained in S_r with the maximum values of $f_{C-\{v'\}}$;
  goto **Loop**;
else if $|S_r| > 0$ **then**
  remove all vertices from $S(C)$ that are contained in S_r;
  goto **Loop**;
else
  return $S(C)$ as $S'(C)$;

4.3.3. Exploratory vertex addition

In our integrative approach, vertex removal and addition operations are alternately executed until the completion of community detection. Here, we explain how *Exploratory-addition*($S(C)$, σ) performs in the process of *Exploratory Vertex Addition*, which is shown in Algorithm 3.

Algorithm 3: Exploratory-addition($S(C)$, σ).

Input: community vertex set $S(C)$, addition coefficient σ;
Output: community vertex set $S''(C)$, Boolean $expandable''(C)$;

integer $threshold \leftarrow 1$;
if $\sigma|S(C)| > 1$ **then**
  $threshold \leftarrow \lfloor \sigma|S(C)| \rfloor$;
Boolean $expandable''(C) \leftarrow TRUE$;
find the neighborhood $N(C)$ of $S(C)$;
foreach $v'' \in N(C)$ **do**
  if $f_C^{v''} > 0$ **then**
    add v'' to the potentially addable vertex set S_a;
if $|S_a| > threshold$ **then**
  add $threshold$ number of vertices to $S(C)$ that are contained in S_a with the maximum values of $f_C^{v''}$;
else if $|S_a| > 0$ **then**
  add all vertices contained in S_a to $S(C)$;
else
  $expandable''(C) \leftarrow FALSE$;
return $S(C)$ as $S''(C)$, $expandable''(C)$;

Similarly, the addition coefficient σ is elaborated exhaustively in Section 4.3.5.

4.3.4. Community fitness calculation

In the *EVRA* process, we perform a number of redundant computations if the fitness function of community C is completely calculated after every time a vertex is added to or removed from C, which leads to the exponential increase of time complexity with the increase of vertices.

A scheme is proposed to avoid massive redundant operations, among which the process of identifying the potentially addable vertices for *Exploratory Vertex Addition* is described in Algorithm 4, and the process of *Exploratory Vertex Removal* can be deduced by analogy.

Algorithm 4: Potentially-addable-vertex-set($S(C)$).

Input: community vertex set $S(C)$;
Output: potentially addable vertex set S_a;

```
find the neighborhood N(C) of S(C);
calculate initial_fitness(C), initial_numerator(C) and initial_denominator(C);
foreach p ∈ N(C) do
    add_numerator = initial_numerator(C);
    add_denominator = initial_denominator(C);
    foreach undirected edge e^undir of p do
        if e^undir connects a vertex v in S(C) then
            add_numerator+ = 1;
            add_denominator+ = 1/2;
        else
            add_denominator+ = 1/2;
    foreach out-directed edge e^out of p do
        if e^out connects a vertex v in S(C) then
            add_numerator+ = 1;
            add_denominator+ = 1;
        else
            add_denominator+ = 1;
    foreach in-directed edge e^in of p do
        if e^in connects a vertex v in S(C) then
            add_numerator+ = 1;
    if add_numerator / add_denominator > initial_fitness then
        add p to the potentially addable vertex set S_a;
return S_a;
```

In Algorithm 4, given a community C, $initial\_fitness(C)$ represents the initial fitness value, $initial\_numerator(C)$ and $initial\_denominator(C)$ are the numerator and denominator values of $initial\_fitness(C)$ separately.

4.3.5. Feasibility theory of our approach

Based on the description of phenomena on community changes in [36], we formalize the community drift as follows.

Definition 5 (Community Drift)**.** Not counting the initial community, after a new round of EVRA, if community C does not retain more than half of the previous vertices, then we call this phenomenon community drift.

Proposition 1. *Assuming that community C contains n_C vertices, we add σn_C vertices selected from the neighborhood to C that maximally increase f_C. If $\sigma \in [0, 1)$, then the community drift does not occur in C.*

Proof of Proposition 1. Adding a single vertex to community C increases the value of f_C, and the new added vertex is compatible with the previous vertices. Thus, no matter how many vertices it holds in C, the community drift does not occur when only one vertex is added to C. In the scenario that σn_C vertices are added to community C, since $\sigma \in [0, 1)$, we obtain $n_C > \sigma n_C$, then $n_C/(n_C + \sigma n_C) > 0.5$, and more than half of the previous vertices in C are retained after the process of Exploratory Vertex Removal, which concludes Proposition 1.

Proposition 2. *Assuming that community C contains n_C vertices, we remove τn_C vertices from C that maximally increase f_C. If the constraint $\tau \in [0, 0.5)$ is satisfied, then the community drift does not occur in C.*

Proof of Proposition 2. When a single vertex is removed from community C, the value of f_C is increased, so the community drift does not occur if only one vertex is removed. The previous vertices in C are compatible with each other, and the new added vertices are compatible with the previous vertices as well. In addition, the number of new added vertices is smaller than that of previous vertices. Herein $\tau \in [0, 0.5)$, so more than half of the previous vertices are retained in C after the process of Exploratory Vertex Removal. That is, Proposition 2 is verified to be true.

After executing the community detection by starting from every seed vertex, if the identified communities fail to cover the entire network, then we carry out the seeds selection on remaining vertices and the related communities detection until all vertices of the network are contained in communities. The seeds selection is based on the local extreme conductance,

and the number of seeds selected in each round increases linearly as the scale of network increases. Therefore the number of rounds of seeds selection is almost independent of the network size. A large number of experiments confirmed that the detected communities can cover the entire network through only a few rounds (usually no more than 10) of seeds selection.

Let us take Fig. 3 as an instance for overlapping community detection, where the user model o belongs to communities B and C simultaneously. We find that the seed vertices c' and c'' belong to the same community C after the *EVRA* process, which means that only one record needs to be kept. A number of experiments on real-world networks revealed that the probability of two or more seeds belonging to the same community is very low.

4.4. Membership degree calculation

The non-overlapping vertex only attaches to a single community with the membership degree equal to 1. However, the overlapping vertex is jointly overlapped in multiple communities. Thus, the membership degrees are unequally distributed in these associated communities, but sum up to 1.

Assume that C_p^i represents the i-th community of overlapping vertex p, and N_p^i indicates the vertex set whose items affiliate to C_p^i and are adjacent to p. In this study, the membership degree of p in C_p^i denoted by β_p^i is defined in the following equations:

$$A^{undir}(p,q) = \frac{1}{2m^{undir}}\left(M_{pq}^{undir} - \frac{d_p^{undir} d_q^{undir}}{2m^{undir}}\right). \tag{16}$$

$$A^{dir}(p,q) = \frac{1}{m^{dir}}\left(M_{pq}^{dir} - \frac{d_p^{out} d_q^{in}}{m^{dir}}\right). \tag{17}$$

$$\beta_p^i = \sum_{q\in N_p^i} \frac{1}{o_p o_q}\left(A^{undir}(p,q) + A^{undir}(q,p) + A^{dir}(p,q) + A^{dir}(q,p)\right). \tag{18}$$

Where $A^{undir}(p, q)$, m^{undir} and M_{pq}^{undir} are the affinity degree between vertex p and q, the number of edges and the adjacent matrix element in the undirected network, respectively; d_p^{undir} is the degree of p attaching to undirected edges; $A^{dir}(p, q)$, m^{dir} and M_{pq}^{dir} are the affinity degree pointing from vertex p to q, the number of edges and the adjacent matrix element in the directed network, respectively; d_p^{out} represents the out-degree of p and d_q^{in} denotes the in-degree of q; and o_p expresses the number of communities of p.

After calculating the initial values, without loss of generality, the sum of membership degrees of each vertex in different communities is then normalized to 1.

4.5. Hierarchical community mining

The communities in the first-level network are regarded as vertices and the relations among these communities are transformed into edges (containing directed and undirected edges) to construct a second-level hierarchical network.

According to [29], the interest similarity between community C_v and C_w, denoted by S_{vw}^{undir}, is derived as

$$S_{vw}^{undir} = \sum_{p\in C_v, q\in C_w, p\neq q} 2A^{undir}(p,q)\delta_N^{undir}(C_v, C_w). \tag{19}$$

Where $A^{undir}(p, q)$ is the affinity degree between vertex p and q in the undirected network, with details shown in Eq. (16); $\delta_N^{undir}(C_v, C_w)$ is an impulse function such that it returns 1 if there exist undirected edges between vertices in C_v and ones in C_w and 0 if else.

The affinity degree pointing from community C_v to C_w, marked with S_{vw}^{dir}, is defined as

$$S_{vw}^{dir} = \sum_{p\in C_v, q\in C_w, p\neq q} A^{dir}(p,q)\delta_N^{dir}(C_v, C_w). \tag{20}$$

Where $A^{dir}(p, q)$ is the affinity degree pointing from vertex p to q in the directed network, with details explained in Eq. (17); $\delta_N^{dir}(C_v, C_w)$ is also an impulse function such that it returns 1 if there exist directed edges pointing from vertices in C_v to ones in C_w and it returns 0 if else.

Assuming that there exist k communities in the first-level network, we then connect undirected edges between $den^{undir}[k(k-1)/2]$ unordered community-pairs with maximum interest similarity values to build the second-level undirected network, and connect directed edges between $den^{dir}(k(k-1))$ ordered community-pairs with maximum affinity degrees of social interaction to construct the second-level directed network. Herein, den^{undir} and den^{dir} are adjustable parameters, determined by the machine learning algorithm based on application requirements, to reflect the resolution ratio of communities.

Afterwards, the overlapping community detection is conducted on the second-level hierarchical network, and so forth, until the entire network becomes some isolated vertices.

4.6. Time complexity analysis

In the *EVRA* process, multiple vertices are added to or removed from the community in each iteration. Experiments have confirmed that through a few (usually no more than 20) iterations of vertices addition or removal operation, every community can reach a steady state. Since the redundant computation is eliminated in the process of calculating community fitness, the time complexity of vertices addition or removal operation is $O(D_i)$, where D_i denotes the sum of the degrees (containing the degrees attaching to undirected edges, in-degrees and out-degrees) of all vertices in the current community. Thus, the time complexity of the final community detection is $O(D'_i)$, where D'_i represents the sum of the degrees of all vertices in the final community. Consequently, the overall time complexity of the overlapping community detection is $O(m)$, where m indicates the number of edges in the network.

In the process of hierarchical community detection, we calculate the correlation coefficient only when edges between the related communities exist. Each community is adjacent to a small number of communities, so the overall time complexity of the hierarchical community detection is $O(n)$, where n represents the number of vertices in the network.

In a connectivity network, since $m > n$, the overall time computational complexity of our OCD-HSN method is $O(m)$.

If the proposed approach runs in a distributed parallel computing environment, since each stage in the process can be executed concurrently, the time complexity can be significantly reduced to $O(m/num_{pc})$, where num_{pc} indicates the number of computers working in parallel. However, in practical applications, the time consumption would be affected by many factors, including communication speed, storage performance, and task assignment.

5. Experiments

In this section, we first introduce four assessment criteria that are extended from traditional unweighted and undirected networks, and two state-of-the-art methods that are employed for comparison with our proposed approach. Second, we present the experimental results on both synthetic and real-world heterogeneous social networks. Finally, we discuss and analyze the experimental results.

5.1. Evaluation metrics

5.1.1. Omega index

The omega index [10] is the overlapping version of the *Adjusted Rand Index* [17]. It is based on the consistencies of vertex-pairs in community detection results to evaluate the performances of methods. Herein, the consistency of a vertex-pair is determined by whether these two vertices belong to exactly the same number of communities.

Supposing there are two different community detection results R_1 and R_2, the omega index is defined as

$$\Omega(R_1, R_2) = \frac{\Omega_u(R_1, R_2) - \Omega_e(R_1, R_2)}{1 - \Omega_e(R_1, R_2)}. \tag{21}$$

Where $\Omega_u(R_1, R_2)$ and $\Omega_e(R_1, R_2)$ calculate the unadjusted omega index and the expected omega index in the null model, respectively.

The closer R_1 and R_2 are, the greater the value of $\Omega(R_1, R_2)$ is.

5.1.2. Precision and recall

Precision, recall and F-score [11,32], widely used in the field of information retrieval, are employed to estimate the overlapping attribute accuracy of vertices in community detection.

Given a vertex u, the related definitions of these three evaluation criteria are formalized as

$$Precision = \frac{|C^u_{R_2} \bigcap C^u_{R_1}|}{|C^u_{R_2}|}. \tag{22}$$

$$Recall = \frac{|C^u_{R_1} \bigcap C^u_{R_2}|}{|C^u_{R_1}|}. \tag{23}$$

$$F = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}. \tag{24}$$

Where $C^u_{R_1}$ and $C^u_{R_2}$ are the community sets that u belongs to in the factual community result R_1 and the community detection result R_2, respectively.

The two communities, which come from R_1 and R_2 separately, are considered equal to each other if the commonality ratio of vertices is greater than the parameter λ [7]. For simplicity, in the process of calculating the precision, for each community in C_{R_2}, we define the communities with a maximum commonality ratio of vertices in C_{R_1} as the equal ones, and vice versa in the process of calculating the recall.

We can obtain the mean precision, recall and F-score of all vertices in R_2 using statistical analysis. The more accurate R_2 is, the greater the precision, recall and F-score are.

5.1.3. Normalized mutual information

Given a set of true modules and the set of clusters detected by an algorithm, these sets of communities must be compared to quantify how similar or different the sets are.

An assessment index based on normalized mutual information was proposed in [26] to evaluate overlapping community finding approaches, which corrects the unintuitive behavior in the version presented in [21].

The mutual information between cover X and Y is defined by

$$I(X:Y) = \frac{1}{2}[H(X) - H(X|Y) + H(Y) - H(Y|X)]. \tag{25}$$

Where $H(X)$ represents the entropy of X and $H(X|Y)$ denotes the variation of information between X and Y.

Since 0 means dissimilarity and 1 implies perfect similarity, the normalized mutual information is recommended as

$$NMI_{max} = \frac{I(X:Y)}{max(H(X), H(Y))}. \tag{26}$$

5.1.4. Modularity

The modularity value of all communities in an unweighted and undirected network [28] is calculated as

$$Q^{undir} = \sum_{i=1}^{n_C} \sum_{p,q \in C^i} A^{undir}(p,q)\beta_p^{C^i}\beta_q^{C^i}. \tag{27}$$

Where n_C is the total number of communities in network; C^i indicates the i-th community; $A^{undir}(p, q)$ denotes the affinity degree between vertex p and q with details shown in Eq. (16); and $\beta_p^{C^i}$ represents the membership degree of p in C^i.

The modularity value of all communities in an unweighted and directed network [23] is derived as

$$Q^{dir} = \sum_{i=1}^{n_C} \sum_{p,q \in C^i} A^{dir}(p,q)\beta_p^{C^i}\beta_q^{C^i}. \tag{28}$$

Where $A^{dir}(p, q)$ is the affinity degree pointing from vertex p to q with details presented in Eq. (17).

In this study, the modularity value in a heterogeneous network is defined as

$$Q = \frac{w^{undir}Q^{undir} + w^{dir}Q^{dir}}{w^{undir} + w^{dir}}. \tag{29}$$

Similarly, without involving any specific application in this study, for the sake of simplicity we set $w^{undir} = 1$ and $w^{dir} = 1$. Likewise, the more accurate the community detection result is, the greater the modularity value is.

5.2. Comparison methods

5.2.1. Gleich's Method

Gleich [15] selects vertices as seeds using the optimum seeking method, and then determines communities with the random walk algorithm [4]. By extending it to heterogeneous networks, the vertices, of which the seed-neighborhoods have conditional local minimum conductance values, are selected as seeds.

The specific process of this community detection method is as follows.

(i) Set the own probability transition parameter $\alpha = 0.99$. Initialize the community to a vertex set, which contains all the vertices in the seed-neighborhood, and the followers of central vertex, of which the seed-neighborhoods have greater conductance values. Set the expected community size ψ equal to the number of vertices in the initial community. In the process of probability transiting between vertices, the weight value of outgoing edges is twice that of undirected edges, with the exclusion of the incoming edges.

(ii) Carry out the PageRank random walk until the translatable probability is smaller than $1/(10\psi)$.

(iii) Sweep over all cuts induced by the ordering of the standardized vertex probabilities, and choose the best vertex set as the final community. The standardized probability of vertex v is calculated as

$$pro'(v) = \frac{pro(v)}{d_v^{undir} + 2d_v^{out}}. \tag{30}$$

Where $pro(v)$ is the initial probability of v; d_v^{undir} and d_v^{out} are the degree attaching to undirected edges and out-degree of v, respectively.

The time complexity is $O(m)$, where m represents the number of edges in the network.

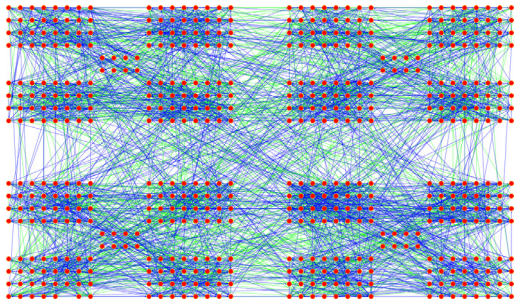


Fig. 5. Artificial network of small size with triple-level structure.

5.2.2. Lancichinetti's method

By extending Lancichinetti's method [21] to heterogeneous networks, we then obtain a new expression: Start from a randomly selected seed vertex to expand the community until the community fitness f_C reaches a steady maximum value. f_C is defined as

$$W_{out}^{C} = W_{out_{undir}}^{C} + W_{out_{dir}}^{C}. \tag{31}$$

$$f_C = \frac{W_{in}^{C}}{(W_{in}^{C} + W_{out}^{C})^{\eta}}. \tag{32}$$

Where $W_{out_{undir}}^{C}$ is half the number of undirected edges connecting vertices in the community to other external ones; $W_{out_{dir}}^{C}$ is the number of directed edges pointing from vertices in the community to other external ones; W_{in}^{C} is the number of edges in the community including both directed and undirected ones; and η is the resolution parameter that determines the community size; in the experiments, we set $\eta = 1$ to maintain fairness.

After one community is completed, the method randomly selects another vertex not yet assigned to any community as the seed to unfold a new community. The algorithm detects the local communities iteratively until every vertex belongs to at least one community.

The time complexity is $O(n^2)$, where n denotes the number of vertices in the network.

5.3. Experimental results on artificial networks

Experiments are conducted on a computer with two four-core 2.33-GHz processors, 16GB RAM, and Windows Server 2008 R2 OS. The implementation is done using the programming language Java 1.7 (64-bit) and the relational database MySQL 5.5 (64-bit).

5.3.1. Artificial networks construction

To verify the validity and perform the scalability analysis of our OCD-HSN approach, we generate four synthetic hierarchical networks that are similar to the benchmark proposed by Arenas et al. [5,6]. The characteristics of these artificial networks are detailed below.

(1) Triple-level network of small size (TL-SS)

There are 544 vertices in total, arranged in 16 communities of 32 non-overlapping and 8 overlapping vertices each in the first-level network, and each overlapping vertex is assigned to 4 communities with the membership degree identically equal to 0.25. These 16 groups are organized into 4 communities containing 136 non-overlapping vertices each in the second-level network. The process repeats until all vertices are assigned to the same single community in the third-level network.

In any community in the first-level network, the connection probabilities between non-overlapping vertices are pro_1^{undir} for undirected edges and pro_1^{dir} for directed edges, and those between any overlapping vertex and the others are $pro_{o_1}^{undir}$ for undirected edges and $pro_{o_1}^{dir}$ for directed edges. The connection probabilities between vertices are required to satisfy

$$pro_1^{undir} + pro_1^{dir} > pro_{o_1}^{undir} + pro_{o_1}^{dir} \geq pro_2^{undir} + pro_2^{dir} > pro_3^{undir} + pro_3^{dir}. \tag{33}$$

In this process, we set $pro_1^{undir} = pro_1^{dir} = 0.1613$, $pro_{o_1}^{undir} = pro_{o_1}^{dir} = pro_2^{undir} = pro_2^{dir} = 0.0185$ and $pro_3^{undir} = pro_3^{dir} = 0.0033$; we then obtain the TL-SS network as shown in Fig. 5.

(2) Quadruple-level network of medium size (QL-MS)

Table 1
Sequential execution time of OCD-HSN with different combinations of addition coefficient σ and removal coefficient τ on the TL-SS network.

Combination		Time (sec)	Combination		Time (sec)	Combination		Time (sec)
σ	τ		σ	τ		σ	τ	
0.1	0.1	736.357	0.4	0.1	484.820	0.7	0.1	485.404
0.1	0.2	729.009	0.4	0.2	483.458	0.7	0.2	484.174
0.1	0.3	730.205	0.4	0.3	484.709	0.7	0.3	484.388
0.1	0.4	730.494	0.4	0.4	485.080	0.7	0.4	485.277
0.2	0.1	565.353	0.5	0.1	489.631	0.8	0.1	491.755
0.2	0.2	563.830	0.5	0.2	483.954	0.8	0.2	484.164
0.2	0.3	566.184	0.5	0.3	484.300	0.8	0.3	484.043
0.2	0.4	566.538	0.5	0.4	485.353	0.8	0.4	484.718
0.3	0.1	505.541	0.6	0.1	487.345	0.9	0.1	485.471
0.3	0.2	500.951	0.6	0.2	484.639	0.9	0.2	485.163
0.3	0.3	501.617	0.6	0.3	485.694	0.9	0.3	485.134
0.3	0.4	502.992	0.6	0.4	486.289	0.9	0.4	484.531

There are 2240 vertices in total, split into 64 overlapping and 4 non-overlapping communities in the first-level network. Specifically, each overlapping community has 32 non-overlapping and 8 overlapping vertices, wherein each overlapping vertex is associated with 4 communities with the membership degree equally distributed, while each non-overlapping community only contains 32 vertices. In the second-level network, these 68 sub-modules are partitioned into 16 communities composed of 136 non-overlapping and 32 overlapping vertices, wherein each overlapping vertex belongs to 4 communities with equal membership degree. These non-overlapping and overlapping vertices are separately contained in 4 overlapping communities and a non-overlapping community in the previous level network. These 16 clusters are agglomerated into 4 communities containing 560 non-overlapping vertices each in the third-level network. The procedure is executed repeatedly until all the vertices are merged into one community in the fourth-level network.

In any community in the second-level network, the link probabilities between any overlapping vertex and any non-overlapping one, which also acts as a non-overlapping vertex in the first-level network, are denoted $pro_{o_2}^{undir}$ for the undirected edge and $pro_{o_2}^{dir}$ for the directed edge, respectively.

In this network, we set $pro_{o_2}^{undir} = pro_{o_2}^{dir} = 0.0185$ and $pro_4^{undir} = pro_4^{dir} = 0.0006$, and the remaining arguments are assigned the same values as in the TL-SS network.

(3) Triple-level network of medium size (TL-MS)

This network is a simple extension of the TL-SS network. It contains 5440 vertices that are distributed into 160 communities in the first-level network, and each community has 32 non-overlapping and 8 overlapping vertices. In particular, each overlapping vertex crosses 4 different communities with a membership degree of equal value. These 160 modules are incorporated into 40 communities of 136 non-overlapping vertices each in the second-level network. The process iterates until all vertices are joined to form the same single community in the third-level network.

We set $pro_3^{undir} = pro_3^{dir} = 0.00033$, and the remaining parameters are given the same values as in the TL-SS network.

(4) Triple-level network of large size (TL-LS)

Analogous to the TL-MS network, there are 54,400 vertices in the entire network, which are ordered into 1600 groups in the first-level network, and there exist 400 communities in the second-level network.

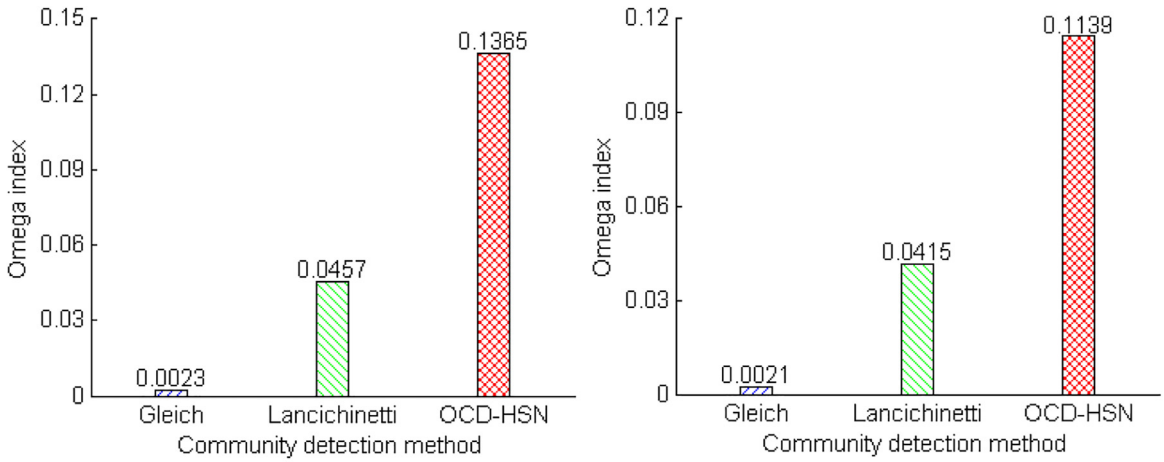
The connection probabilities are set to the same values as in the TL-SS network, except two parameters, which are assigned as $pro_3^{undir} = pro_3^{dir} = 0.000033$.

5.3.2. Results on artificial networks

Theoretical analysis and experimental data indicate that the variations of addition coefficient $\sigma \in [0, 1)$ and removal coefficient $\tau \in [0, 0.5)$ of our proposed approach will not affect the final result of community detection, but can impact the execution efficiency. As illustrated in Table 1, the amount of sequential computing time needed for community detection on the TL-SS network varies with the different combination of σ and τ values. At point $\sigma = 0.4$ and $\tau = 0.2$, the algorithm reaches the highest efficiency. Thus, these two parameter values are referred to as default configurations in the following comparative experiments.

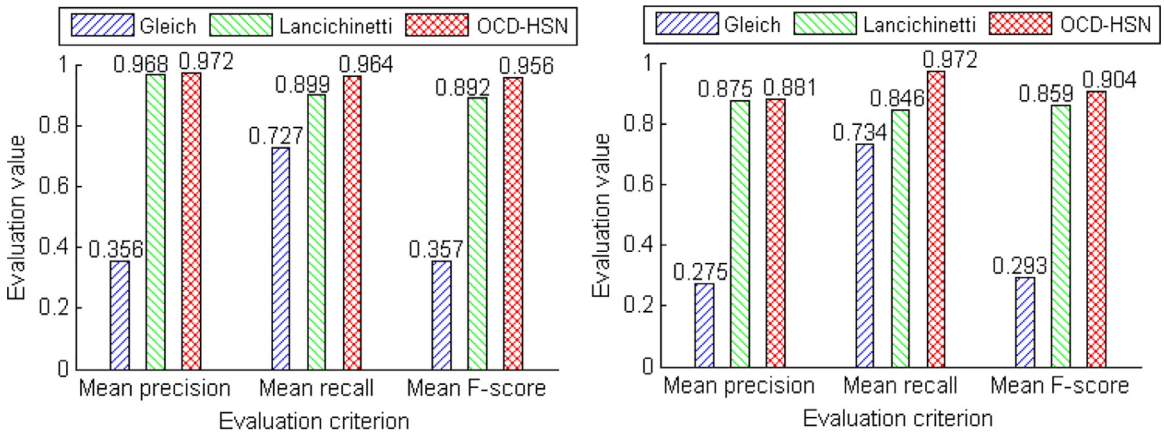
Based on the known topology structures of artificial networks, the factual community structures are treated as references to compare the omega index values of different community finding methods, and the results on the TL-SS and QL-MS networks are shown in Fig. 6. One can notice that the omega indexes of OCD-HSN are much higher than those of the two compared approaches.

Similarly, the factual module structures are regarded as reference standards to compare the mean precision, recall and F-score values of different community detection algorithms, and the results on the TL-SS and QL-MS networks are displayed in Fig. 7. From the results, we can conclude that our scheme outperforms the other approaches in all three aspects.



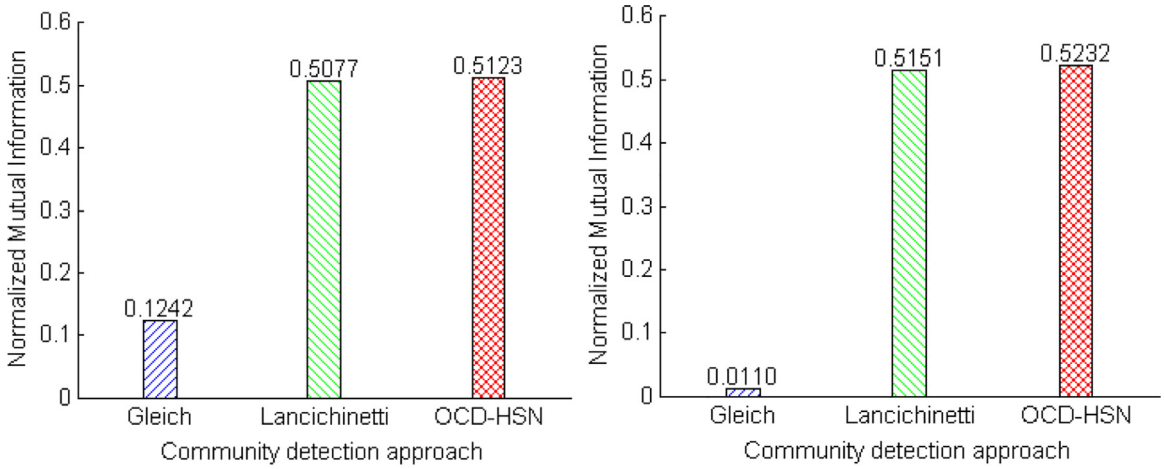
(a) TL-SS (b) QL-MS

Fig. 6. Omega index values of different community detection algorithms on two artificial networks.



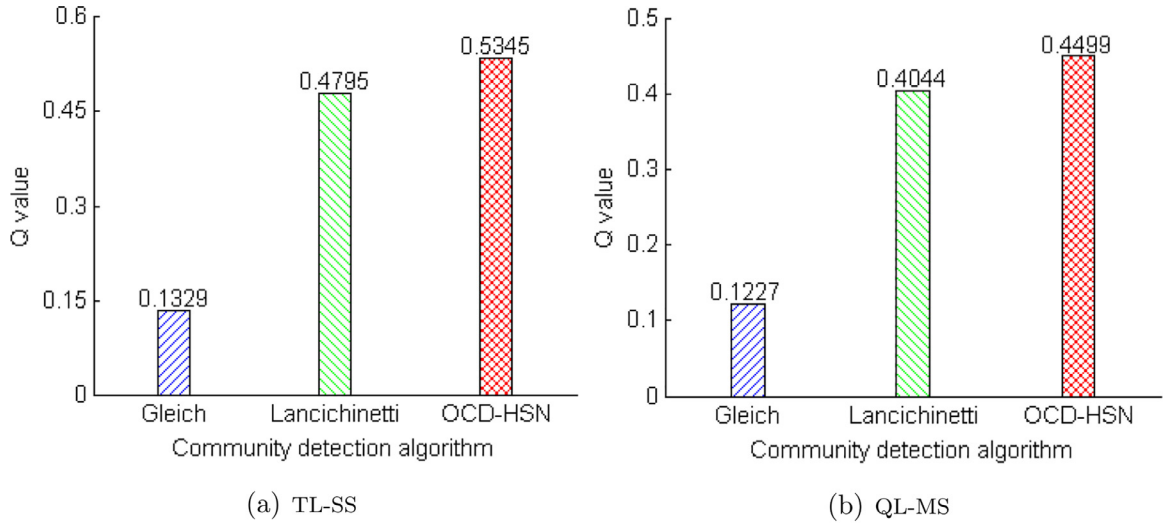
(a) TL-SS (b) QL-MS

Fig. 7. Mean precision, recall and F-score values of different community detection methods on two artificial networks.



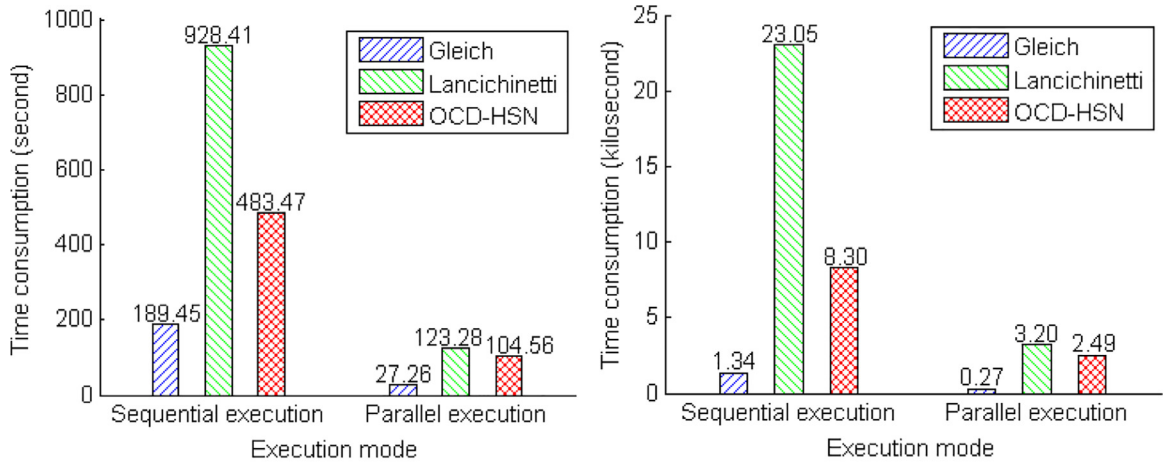
(a) TL-SS (b) QL-MS

Fig. 8. Normalized mutual information values of different community detection algorithms on two artificial networks.



(a) TL-SS (b) QL-MS

Fig. 9. Modularity Q function values of different community detection schemes on two artificial networks.



(a) TL-SS (b) QL-MS

Fig. 10. Execution time of different community detection algorithms on two artificial networks.

Fig. 8demonstrates the normalized mutual information values of these three comparison schemes on the TL-SS and QL-MS networks. Our method achieves the highest performance despite the fact that it only has a small advantage over Lancichinetti's approach.

We compare Newman's modularity Q function values of different community mining algorithms on the TL-SS and QL-MS networks. As presented in Fig. 9, the approach proposed in this study gains the highest performance.

In order to compare the execution efficiencies of different community detection methods on the TL-SS and QL-MS networks, we perform sequential and parallel computing with 15 threads, and obtain the results as displayed in Fig. 10. In the parallel computing processes, for the sake of fairness, the Lancichinetti's method iteratively selects 15 random vertices not yet assigned to any identified cluster as seeds to expand new communities.

During the hierarchical community detection, the resolution ratio of communities is determined by the combinatorial values of den^{undir} and den^{dir}. We set $pro_x^{undir} = pro_x^{dir}$ in the artificial network constructions, where x represents a variable symbol with details shown in Eq. (33), so we also set $den^{undir} = den^{dir}$ in the related experiments. As illustrated in Fig. 11, the resolution ratio of communities at multiple levels of hierarchy varies with different values of den^{undir} and den^{dir}. The experiments show that the community detection results tally with the actual situation when the network densities are bounded in the interval values [0.2, 0.3], [0.08, 0.1] and [0.3, 0.7] for the second-level of TL-SS and the second and third levels of QL-MS, respectively. To keep information more complete in the process, we select 0.1, the maximum value in the interval, as the network density of the second-level of QL-MS to generate the third-level network.

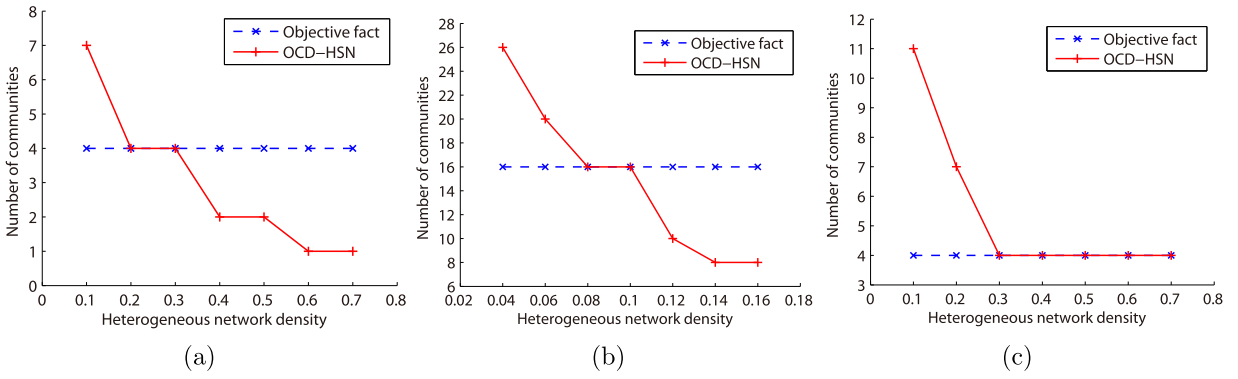


Fig. 11. Relation between the number of communities and the heterogeneous network density in hierarchical community detection on two artificial networks. (a) Result at the second-level of TL-SS. (b) Result at the second-level of QL-MS. (c) Result at the third-level of QL-MS.

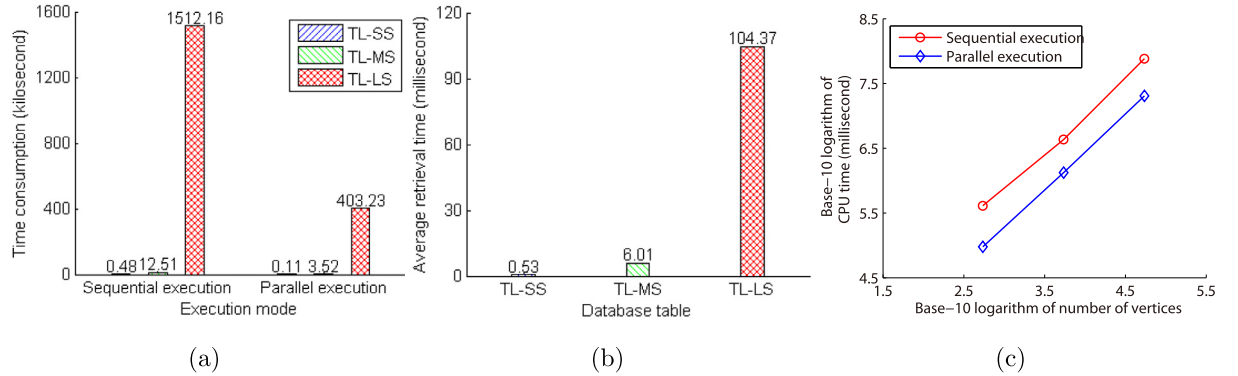


Fig. 12. Experimental results of the linear relation between the CPU time and the number of vertices with approximate constant degree using the OCD-HSN for community detection. (a) Respective total running time on three artificial networks. (b) Separate average time retrieving one record from different database tables. (c) Relation between the CPU time of community detection and the number of vertices in network after taking the logarithm.

To validate the scalability of our OCD-HSN approach, the corresponding computer program is run on the TL-SS, TL-MS and TL-LS networks. At first glance, the time consumption increases in an exponential manner along with the increasing number of vertices, as depicted in Fig. 12(a). This phenomenon occurs due to the limited storage performance. The deep reason is that the time taken for retrieving one record from the database table of TL-LS is 197 times longer than that of TL-SS, as illustrated in Fig. 12(b). Subtracting the access time from the total running time, the CPU time and the number of vertices appear to have an excellent linear relationship, which is certified by the differential coefficient approximate to 1 of their growth relationship after taking the logarithm, as shown in Fig. 12(c). In particular, the vertex degree in these three networks approximately equals a constant value, which means that the CPU time is linear with the number of edges in network, so that the theoretical analysis in Section 4.6 is actually confirmed.

5.4. Experimental results on real-world networks

5.4.1. Real-world social networks construction

We now conduct experiments on three real-life networks for the performance comparison between our OCD-HSN approach and two other community detection methods. The characteristics of these three real-world networks are presented as follows.

(1) Heterogeneous network of Tencent microblogging (HN-TM)

Launched by Tencent Co. Ltd., Tencent microblogging is an online social networking service similar to Twitter. Currently the microblogging users access the service via the web, mobile phone, QQ client, QQ space, e-mail and other means. By the fourth quarter of 2012, the number of registered users reached more than 540 million. At the same time, the number of daily active users has exceeded 100 million. In addition, Tencent microblogging provides a free API [2] for developers to retrieve the global user data. Given these facts, Tencent microblogging is chosen as the experimental platform for the test on a heterogeneous social network.

On average, every user has dozens of followees on the Tencent microblogging platform. If we randomly select some users to build a heterogeneous network, it will neither be representative of its user population, nor accurately reflect the social

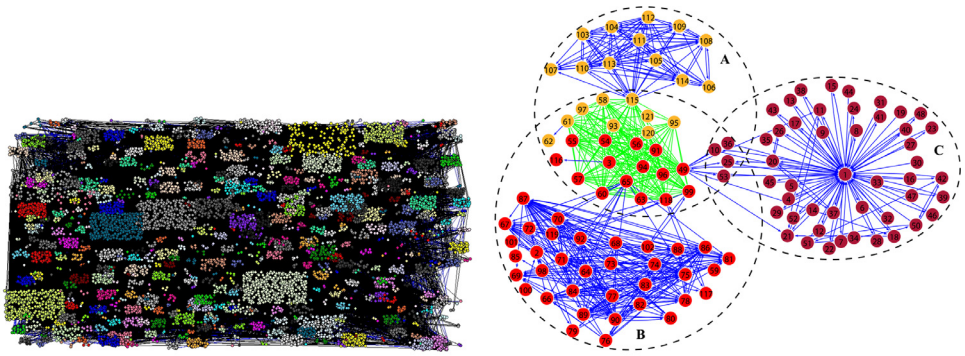


Fig. 13. Community detection result of OCD-HSN on the HN-TM network. (a) Total of 752 detected communities represented by different colors. (b) Visualization of three overlapping communities extracted from the library.

attributes of its users. Therefore, we start from a specified user who is chosen as the initial vertex of the heterogeneous social network, and we then add all followers of this user to the network. In the same way, all the followers of these newcomers are then added, and we repeat the process until the number of vertices reaches the experimental requirement. In addition, those users having more than 1000 followers or followees are excluded, since we regard them as stars or followee abusers.

We conduct the experiment with the data of 12,760 users collected in June 2015. We first model these users respectively using the approach mentioned in Section 3.1, and then construct the heterogeneous social network with the method described in Section 3.2. There exist 83,809 follower-followee records among these users, and for convenience we select 85,492 user-pairs with maximum interest similarity values to construct the user interest network.

(2) Undirected network of Facebook circles (UN-FC)

We build this network using the social circle entries from the Stanford SNAP datasets [24]. Vertices in the network represent Facebook users and undirected edges denote the corresponding friend relationships among them. There are 4039 vertices and 88,234 edges in total.

(3) Directed network of Wikipedia voting (DN-WV)

In order for a user to become a Wikipedia administrator, a vote for adminship is issued to decide who to promote to the position.

All the Wikipedia voting data from the inception of Wikipedia till January 2008 is contained in this network. A directed edge from vertex u to v indicates that user u voted on v. The dataset is also accessible through the Stanford SNAP library [24]. The network is composed of 7115 vertices and 103,689 edges.

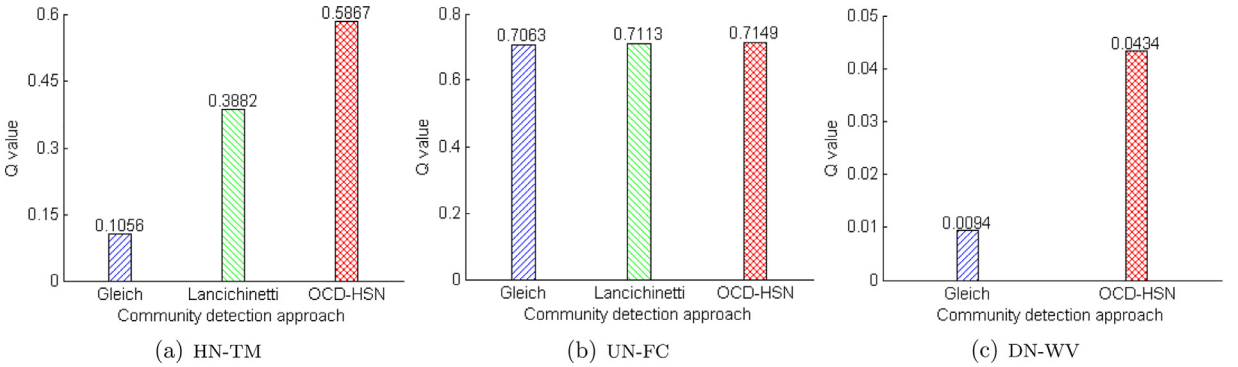
5.4.2. Results on real-world social networks

As displayed in Fig. 13(a), we mine a total of 752 communities by our proposed approach on the HN-TM network. For a much clearer visualization of modular structure, we select three overlapping communities from the community set, as illustrated in Fig. 13(b). There exist 120 vertices in all, among which 24 are overlapped by two clusters and 4 belong to the three communities simultaneously. The overlapping vertex is colored by the community to which the membership value is maximum (in case of a tie, we use a breaking rule).

Without knowing the ground truth of community structure in real-world social networks, we cannot apply the evaluation criteria omega index, normalized mutual information and F-score in this scenario. However, the modularity Q function overcomes this application drawback, and is employed to evaluate the experimental results. As demonstrated in Fig. 14, we calculate the Q values of different community mining algorithms on real-life social networks, and it shows that our approach offers the highest quality.

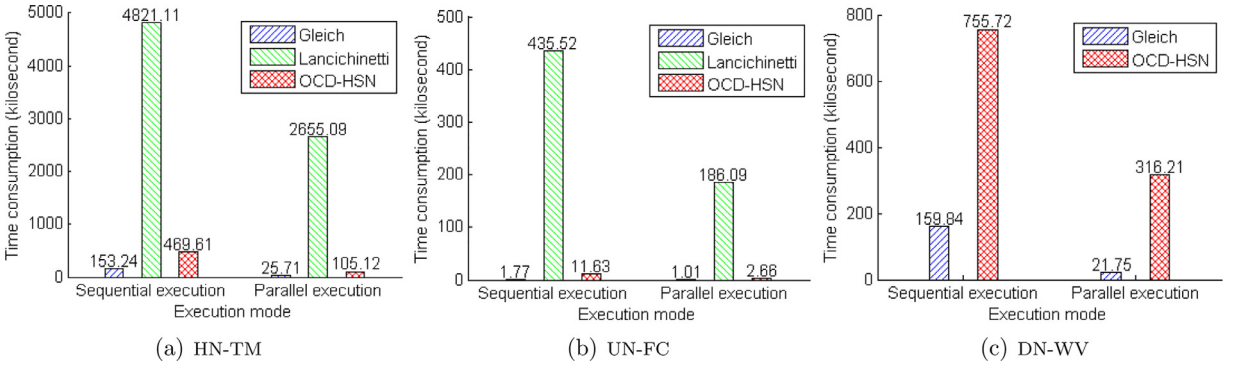
To compare the performance on time consumption among three community finding approaches on real-world social networks, we again execute sequential and parallel computing with 32 threads separately. The experimental results are presented in Fig. 15. Similarly, in the parallel computing stage, Lancichinetti's method randomly selects 32 vertices that do not yet belong to any detected module as seeds to mine new communities. The experimental results show that our scheme is significantly superior to Lancichinetti's algorithm, although it is slightly slower than Gleich's approach.

Specifically, the result of Lancichinetti's approach on the DN-WV network is omitted since we cannot complete the assessment within 30 days even in parallel because of the sluggish rate.



(a) HN-TM (b) UN-FC (c) DN-WV

Fig. 14. Modularity Q function values of different community detection methods on three real-world social networks.



(a) HN-TM (b) UN-FC (c) DN-WV

Fig. 15. Execution time of different community detection algorithms on three real-world social networks.

5.5. Discussion and analysis

Based on the above experimental results, we further discuss and analyze the comparative results between our approach and the other two methods.

For Gleich's method, if the central vertex of a community follows that of another community, these two modules may be forced to merge into one, which violates the regulation of community structure. However, through initializing the community with a seed-neighborhood and executing the removal operation, we can avoid the manmade mistakes of community merging in our approach. Thus, the community accuracy can be significantly improved. In the seed selection processes, experimental results show that our scheme needs only several more rounds than that of Gleich's method to complete community detection. Moreover, our method adds or removes multiple vertices in each iteration. All these make the execution efficiency approach that of a random walk, which has been adopted in Gleich's method.

For Lancichinetti's algorithm, it initializes the community with a randomly selected vertex, failing to take full advantage of the property that the connections between vertices are dense inside but sparse outside communities. On the contrary, our method initializes the community with a seed-neighborhood. The experimental results indicate that the vast majority of these vertices still remain in the final community, which remarkably improves the speed of community detection. Lancichinetti's method adds or removes only one vertex in each iteration during community expansion. By applying the election theory that the majority determines the nature, our approach adds or removes multiple vertices in each iteration on the premise that community drift does not occur, which further improves the execution efficiency. In the aspect of accuracy, since Lancichinetti's method selects seeds randomly, thus when the community fringe vertices are selected as initial communities, it may lead to small unorganized communities, which reduces the community accuracy.

6. Conclusions and Future Work

In this study, we propose an overall framework for overlapping community detection in the heterogeneous social network. First, we construct a heterogeneous social network using the similarity calculation of multiple user interests and the consideration of social interaction relations, where the user's interests and social features are depicted by a user model as a vertex in the network, to provide a new perspective for social computing. We then present an integrative approach for community detection that features three core steps: multiple seeds selection by the vertex set conductance, community initialization by a seed-neighborhood, and multiple vertices addition and removal in each expansion by the election principle.

To compare the accuracy and efficiency of our proposed approach with other existing state-of-the-art methods, we conduct large-scale experiments on both artificial and real-world social networks. The experimental results demonstrate that our scheme outperforms the comparison methods in all aspects of four evaluation metrics, and its efficiency can satisfy the application requirements of large-scale SNSs.

As for the future work, we plan to further extend the strategy of multiple seeds selection for more efficient community detection and continue validating the performance of our approach on more real-life heterogeneous social networks.

Acknowledgments

This work is partially funded by the National Key Research and Development Program of China (No. 2017YFC0907505), the National Natural Science Foundation of China (Nos. 61772128 and 61303096), the Fundamental Research Funds for the Central Universities (No. 16D111208), the Shanghai Natural Science Foundation (No. 17ZR1400200), and the Xinjiang Social Science Foundation (No. 2015BGL100).

References

[1] Baidu encyclopedia, http://baike.baidu.com/.
[2] Tencent microblog open platform, http://dev.t.qq.com/.
[3] V. Agarwal, K.K. Bharadwaj, A collaborative filtering framework for friends recommendation in social networks based on interaction intensity and adaptive user similarity, Soc. Netw. Anal. Min. 3 (3) (2013) 359–379.
[4] R. Andersen, F. Chung, K. Lang, Local graph partitioning using pagerank vectors, in: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06), 2006, pp. 475–486.
[5] A. Arenas, A. Díaz-Guilera, C.J. Pérez-Vicente, Synchronization processes in complex networks, Phys. D 224 (1) (2006) 27–34.
[6] A. Arenas, A. Díaz-Guilera, C.J. Pérez-Vicente, Synchronization reveals topological scales in complex networks, Phys. Rev. Lett. 96 (11) (2006) 114102.
[7] S. Asur, S. Parthasarathy, D. Ucar, An event-based framework for characterizing the evolutionary behavior of interaction graphs, ACM Trans. Knowl. Discov. Data (TKDD) 3 (4) (2009) 16.
[8] V.D. Blondel, J.L. Guillaume, R. Lambiotte, et al., Fast unfolding of communities in large networks, J. Stat. Mech: Theory Exp. 10 (2008) P10008.
[9] S. Calegari, G. Pasi, Personal ontologies: generation of user profiles based on the YAGO ontology, Inf. Process Manage 49 (3) (2013) 640–658.
[10] L.M. Collins, C.W. Dent, Omega: a general formulation of the rand index of cluster recovery suitable for non-disjoint solutions, Multivariate Behav. Res. 23 (2) (1988) 231–242.
[11] S.G. Esparza, M.P. O'Mahony, B. Smyth, Mining the real-time web: a novel approach to product recommendation, Knowl. Based Syst. 29 (2012) 3–11.
[12] S. Fortunato, Community detection in graphs, Phys. Rep. 486 (3) (2010) 75–174.
[13] G. Ghoshal, G. Mangioni, R. Menezes, et al., Social system as complex networks, Soc. Netw. Anal. Min. 4 (1) (2014) 1–2.
[14] M. Girvan, M.E.J. Newman, Community structure in social and biological networks, Proc. Natl. Acad. Sci. 99 (12) (2002) 7821–7826.
[15] D.F. Gleich, C. Seshadhri, Vertex neighborhoods, low conductance cuts, and good seeds for local community methods, in: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge discovery and data mining, 2012, pp. 597–605.
[16] Y. Gu, B. Zhang, G. Zou, et al., Overlapping community detection in social network based on microblog user model, in: Proceedings of the 2014 IEEE International Conference on Data Science and Advanced Analytics (DSAA 2014), 2014, pp. 333–339.
[17] L. Hubert, P. Arabie, Comparing partitions, J. Classification 2 (1) (1985) 193–218.
[18] K. Ikeda, G. Hattori, C. Ono, et al., Twitter user profiling based on text and community mining for market analysis, Knowl. Based Syst. 51 (2013) 35–47.
[19] J. Kamahara, T. Asakawa, S. Shimojo, et al., A community-based recommendation system to reveal unexpected interests, in: Proceedings of the 11th IEEE International Conference on Multimedia Modelling (MMM 2005), 2005, pp. 433–438.
[20] I.A. Kovács, R. Palotai, M.S. Szalay, et al., Community landscapes: an integrative approach to determine overlapping network module hierarchy, identify key nodes and predict network dynamics, PLoS ONE 5 (9) (2010) e12528.
[21] A. Lancichinetti, S. Fortunato, J. Kertész, Detecting the overlapping and hierarchical community structure in complex networks, New J. Phys. 11 (3) (2009) 033015.
[22] D. Lazer, A.S. Pentland, L. Adamic, et al., Life in the network: the coming age of computational social science, Science (New York, NY) 323 (5915) (2009) 721.
[23] E.A. Leicht, M.E.J. Newman, Community structure in directed networks, Phys. Rev. Lett. 100 (11) (2008) 118703.
[24] J. Leskovec, A. Krevl, SNAP Datasets: Stanford large network dataset collection, http://snap.stanford.edu/data/, 2014.
[25] R. Mall, R. Langone, J.A.K. Suykens, Multilevel hierarchical kernel spectral clustering for real-life large scale complex networks, PLoS ONE 9 (6) (2014) e99966.
[26] A.F. McDaid, D. Greene, N. Hurley, Normalized mutual information to evaluate overlapping community finding algorithms, (2011). arXiv preprint arXiv: 1110.2515.
[27] K. Mei, B. Zhang, J. Zheng, et al., Method of recommend microblogging based on user model, in: Proceedings of the 12th IEEE International Conference on Computer and Information Technology (CIT 2012), 2012, pp. 1056–1060.
[28] M.E.J. Newman, M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69 (2) (2004) 026113.
[29] V. Nicosia, G. Mangioni, V. Carchiolo, et al., Extending the definition of modularity to directed graphs with overlapping communities, J. Stat. Mech. 03 (2009) P03024.
[30] S. Papadopoulos, Y. Kompatsiaris, A. Vakali, et al., Community detection in social media, Data Min. Knowl. Discov. 24 (3) (2012) 515–554.
[31] G. Salton, M.J. McGill, Introduction to modern information retrieval, 1986.
[32] B. Sarwar, G. Karypis, J. Konstan, et al., Analysis of recommendation algorithms for e-commerce, in: Proceedings of the 2nd ACM Conference on Electronic Commerce, 2000, pp. 158–167.
[33] S.E. Schaeffer, Graph clustering, Comput. Sci. Rev. 1 (1) (2007) 27–64.
[34] H. Shen, X. Cheng, K. Cai, et al., Detect overlapping and hierarchical community structure in networks, Physica A 388 (8) (2009) 1706–1712.
[35] J. Xie, S. Kelley, B.K. Szymanski, Overlapping community detection in networks: the state-of-the-art and comparative study, ACM Comput. Surv. (csur) 45 (4) (2013) 43.
[36] Z. Zhang, D. Miao, J. Qian, Detecting overlapping communities with heuristic expansion method based on rough neighborhood, Chinese journal of computers 36 (10) (2013) 2078–2086.
[37] J. Zheng, B. Zhang, X. Yue, et al., Neighborhood-user profiling based on perception relationship in the micro-blog scenario, Web Semant. 34 (2015) 13–26.