Knowl Inf Syst (2012) 33:577–601
DOI 10.1007/s10115-012-0540-7

REGULAR PAPER

Efficient algorithms for influence maximization in social networks

Yi-Cheng Chen · Wen-Chih Peng · Suh-Yin Lee

Received: 3 November 2011 / Revised: 25 February 2012 / Accepted: 11 August 2012 /
Published online: 5 September 2012

Abstract In recent years, due to the surge in popularity of social-networking web sites, considerable interest has arisen regarding *influence maximization* in social networks. Given a social network structure, the problem of influence maximization is to determine a minimum set of nodes that could maximize the spread of influences. With a large-scale social network, the efficiency and practicability of such algorithms are critical. Although many recent studies have focused on the problem of influence maximization, these works in general are time-consuming when a social network is large-scale. In this paper, we propose two novel algorithms, CDH-Kcut and Community and Degree Heuristic on Kcut/SHRINK, to solve the influence maximization problem based on a realistic model. The algorithms utilize the community structure, which significantly decreases the number of candidates of influential nodes, to avoid information overlap. The experimental results on both synthetic and real datasets indicate that our algorithms not only significantly outperform the state-of-the-art algorithms in efficiency but also possess graceful scalability.

Keywords Community discovery · Diffusion models · Influence maximization · Social network

1 Introduction

In the last decade, social network analysis has drawn much attention due to its widespread applicability. A social network is a social structure made up of individuals who are tied by one

Y.-C. Chen (✉) · W.-C. Peng · S.-Y. Lee
Department of Computer Science, National Chiao Tung University,
Hsinchu 300, Taiwan
e-mail: ejen.cs95g@nctu.edu.tw

W.-C. Peng
e-mail: wcpeng@cs.nctu.edu.tw

S.-Y. Lee
e-mail: sylee@csie.nctu.edu.tw

or more specific types of relationship or interdependency, such as friendship, co-authorship, common interest, or financial exchange, to name a few. Nowadays, many worldwide social-networking web sites, such as Facebook and Twitter, are very popular since users can share their thoughts and comments with their friends and also bring small and disconnected social networks together. In 2011, Facebook and Twitter already had more than 600 million and about 90 million active users, respectively. Hence, marketing on online social networks shows great potential to be much more successful than traditional marketing techniques. In many enterprises, the budget of advertisement spending on worldwide social-networking sites is almost the same or even in excess of that spent in traditional ways.

For example, a company develops a software "cooler" and wants to market it to a social network. The company has a limited budget so it can only give the free "cooler" to a small number of initial users. The company hopes that the initial users could influence their friends to use the product, and their friends could influence their friends' friends. Through the word-of-mouth effect, the company makes a large number of users adopt the "cooler." The *influence maximization* problem [7] aims to select initial users (referred to as *seeds*) so that the number of users that adopt the product or innovation is maximized. That is, the problem is how to find the influential individuals in a social network.

Formally speaking, a social network is generally modeled as an undirected graph $G(V, E)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is the vertex set and $E = \{(v_i, v_j) |$ there is an edge from v_i to $v_j\}$ is the set of edges. A node represents an individual, and an edge between two nodes represents some kind of relationship (friendship or co-authorship, etc.). A node is marked as active if it has adopted an idea or an innovation, or as inactive if it has not. Thus, the problem of the influence maximization is given below:

> **(Influence Maximization Problem)** Given a social network $G = (V, E)$, the output is to determine a set of seeds (i.e., nodes) such that these seeds could spread their influence to other nodes with the purpose of maximizing the number of nodes affected by the seeds.

Since many social networks become large-scale, developing efficient algorithms for the influence maximization is more and more critical. Kempe et al. [14] proved that the influence maximization problem is *NP-hard*. If it takes a week for companies to decide which set of individuals should be given free samples to promote their products, they may lose their superiority because of non-timeliness. Moreover, the selected set of individuals will not be useful since the network may change significantly during this week. As such, timeliness is an important issue for the influence maximization problem. Some efficient approximate algorithms have been proposed [4,10]; however, although efficient, they are only appropriate for some diffusion models that are not realistic for modeling influences in social networks. Different social networks may have different types of influences. For example, sometimes we want to know which set of individuals could trigger more adoptions of products after 3 days, 7 days, or a month. How to model the influences in social networks is very important. Hence, the diffusion model of influence is another issue for the influence maximization problem. A realistic diffusion model is essential for making correct predictions of the future behavior of the network.

In this paper, we propose a Community and Degree Heuristic approach (CDH) concept to tackle the issues of time efficiency and realistic modeling of the influence maximization problem. According to the CDH concept, two novel algorithms, CDH-Kcut and Community and Degree Heuristic on Kcut/SHRINK (CDH-SHRINK), are developed based on a realistic model. From observation, a common property of social networks is that nodes tend to cluster together. CDH detects the community in social networks to avoid overlap of influence spread

and then selects influential individuals, taking into account the community information by modified degree centrality. To the best of our knowledge, CDH is the first algorithm that combines the technique of community characteristics and modification of degree centrality. Furthermore, experimental studies on both synthetic and real datasets indicate that the proposed CDH is not only efficient but also has good influence spread, that is, the number of influenced nodes.

The rest of this paper is organized as follows. Section 2 introduces the related works on the influence maximization problem. Section 3 details the framework of CDH and the two proposed algorithms, CDH-Kcut and CDH-SHRINK. Section 4 presents the experiments on several synthetic and real datasets. Finally, we conclude and describe the future works in Sect. 5.

2 Related works

2.1 Diffusion models

Rogers [20] theorizes that diffusion is the process by which an innovation is communicated through certain channels over time among the members of a social system. Diffusion is a type of communication concerned with the spread of messages that are perceived as new ideas. Besides, innovations spread through a society as the early adopters select the technology first, followed by the majority, until a technology or innovation becomes popular. Recently, some studies [10,12,17,23,29] have investigated the diffusion models in social networks.

One naïve diffusion model is the linear threshold model (LTM). For an undirected graph $G(V, E)$, $N(v) = \{u|(u, v) \in E\}$ is defined as the neighbor set of node v, and b_{uv} is defined as the influence of active node u on its inactive neighbor v. We define $A(v)$ as the set of active nodes in $N(v)$, $(A(v) \subseteq N(v))$. Given an activation threshold θ, for a node v, if $\Sigma_{u \subseteq A(v)} b_{uv} \geq \theta$, node v becomes active. The intuitive meaning is that for an inactive node v, if the total influence exerted by all its active neighbors exceeds a pre-defined activation threshold θ, node v becomes active. In turn, the newly active node v will exert influence on its inactive neighbors and may make some inactive neighbors become active. This process will continue until no node can be activated.

Another fundamental diffusion model is the independent cascading model (ICM) [12]. If a node u is activated at step t, it will try to activate its inactive neighbor v with success probability p. If it is successful, then v will be active in step $t + 1$, else u is failed and will no longer have a chance to activate v. In addition, each active node has only one chance to activate its neighbor v.

The heat diffusion model (HDM) [17] is a realistic model to simulate social behavior. Heat diffusion is a physical phenomenon. Heat always flows from a position with higher temperature to a position with lower temperature. The phenomenon is similar to the process of people influencing others. The innovators and early adopters of a product or innovation act as heat sources and process a very high amount of heat. These people start to influence others and diffuse their influence to the early majority, then the late majority. In the HDM, the value $f_i(t)$ describes the heat at node v_i at time t, beginning from an initial distribution of heat given at time zero. Suppose at time t, each node v_i receives an amount of heat from its neighbor v_j during a period Δt. The heat should be proportional to the time period Δt and the heat difference $f_j(t) - f_i(t)$. As a result, the heat difference at node v_i between time t and $t + \Delta t$ will be equal to the sum of the heat that it receives from all its neighbors. This is formulated as $\frac{f_i(t+\Delta t)-f_i(t)}{\Delta t} = \alpha \Sigma_{j:(vj,vi)\in E}(f_j(t) - f_i(t)) = \alpha H f(t)$ where H is

a matrix and α is the heat diffusion coefficient. The HDM can easily simulate time effects on information and different types of information flow since non-activated nodes can still spread information. If the amount of heat of node v exceeds the activation threshold θ, we think node v will purchase a product or adopt an innovation.

In reality, different social networks have different information flows. Information on popular web sites transfers faster than on other types of social networks. The time aspect needs to be considered when modeling social network marketing since different marketing strategies are required for information of different durations. It is not reasonable that only activated nodes could spread information. ICM and LTM are built at a very coarse level, typically with only a few global parameters, and are not useful for making correct predictions of the future behavior of the network [6,8]. HDM provides more parameters to simulate the conditions of the real world, such as time and thermal conductivity. Some other models [10,23,29] have been proposed, but all of them are variations of the two core models, LTM and ICM.

Note that a person's decision to buy a product is often strongly influenced by his/her friends and acquaintances. Clearly, the influence maximization problem is how we select the most influential early adopters. Better early adopters could affect more people to adopt the product. Online social networks provide good opportunities to address this problem, since we can easily share information with our friends. Influence maximization problems under LTM, ICM, and HDM are all NP-problems, as already proved in [14,17].

2.2 Influence maximization algorithms

Since the influence maximization problem is an NP-problem, many works have been proposed to achieve approximate solutions. In social networks, we often consider the person who has the most friends as the most influential person, since s/he can possibly influence the most people. Therefore, the intuitive strategy, in general, is to select seeds based on their degree, called *degree centrality*. Nevertheless, the members of large communities often have a larger degree than other members of smaller communities. Consequently, degree centrality can easily select seeds in the same large community. The influence spreads (i.e., the number of influenced nodes) of each seed in the same community tend to overlap. As a result, degree centrality does not have good performance in terms of influence spread. *Distance centrality* is another commonly used method for the influence maximization problem. It selects seeds in the order of increasing average distance to other nodes. However, nodes in larger communities usually have a smaller average distance. As a result, most seeds may also be clustered. Simply stated, degree centrality and distance centrality result in the phenomenon of seed clustering, which leads to a sharp deterioration in influence spread.

The set cover greedy algorithm [10] was developed based on the ICM model. It keeps selecting the node with the highest "uncover degrees." Once a node is selected, all its neighbors as well as itself are labeled as "covered." This procedure continues until k seeds are selected. This algorithm is computationally fast under simpler models, such as ICM. However, it has good influence spread only with high success probability. The climbing-up greedy algorithm [14] was proposed based on the ICM and LTM models with approximation guarantees for influence spread. It selects the most "influential" node on the condition of considering all the seeds selected before. For selecting the most influential node, we have to compute each node's influence until the required k seeds are selected. Due to the heavy computing load, the climbing-up greedy algorithm is not appropriate for large social networks.

Borrowing the idea from [14], the enhanced greedy algorithm [17] was proposed based on the HDM. As with the climbing-up greedy algorithm, we cannot solve the influence maximization problem under the HDM in acceptable time. The potential-based node selection

method [25] was proposed to select some inactive nodes that might not be optimal in the starting phase but that could trigger more nodes in a later stage of diffusion. It can save half the time of the *climbing-up greedy algorithm* and cause more adoptions than that the method in [14]. However, in practice, it is still not efficient enough in on-line social networks. Based on the variation of ICM, Saito et al. [22] constructed a layered graph approach and applied bond percolation with two control strategies, pruning and burnout, to solve the influence maximization problem. The pruning method is effective when searching for a single influential node, and the burnout method is powerful in searching for multiple nodes that together are influential.

The extremely efficient algorithm, the *degree discount heuristic*, was presented in [4,25]. It obtains the approximate solutions in large datasets in only a few seconds. However, both [4,25] are only under LTM or ICM, which are not very realistic diffusion models. In addition, the degree discount heuristic is only for very low successful probability, that is, it is extremely hard for people to be influenced with very low success probability.

2.3 Community structure

A community is characterized as a subset of individuals who interact with each other more frequently than other individuals outside the community [26]. Community discovery is similar but not equivalent to the conventional graph partitioning problem. Both community discovery and the conventional graph partitioning problem aim to cluster vertices into groups. A key challenge for the former, however, is that the algorithm has to decide what is "the best" or, in other words, the "most natural" partition of a network. The "most natural" partitioning method means that we need not give any heuristic information. Furthermore, if there is no good community structure, then there is no need to partition the network. That is why we use the community detection algorithm rather than the conventional graph partitioning algorithm.

A quantitative measure, called modularity (Q), has been proposed [24] to assess the quality of community structures, and community discovery was formulated as an optimization problem. Since optimizing Q is an NP-problem, several heuristic methods have been proposed, as surveyed in [5]. Assume M is the number of edges and N is the number of nodes. The time complexity of most community detection algorithms [3,5,9,11,13,15,18,19,21,24,27,28] is between $O(N \log N)$ and $O(N^3)$. In this paper, the efficiency of the algorithms is our greatest concern, so we select Kcut [21] and SHRINK [13], which have low time complexity $O(M \log N)$ and good modularity, as our community detection algorithms. The SHRINK algorithm can also detect hubs that are very useful for the influence maximization problem.

3 Algorithm CDH

In this section, we first present the design of the CDH. Then, based on the community detection algorithms used (i.e., Kcut and SHRINK), we propose two algorithms, CDH-Kcut and CDH-SHRINK.

3.1 Concept overview

Given a social network G with N individuals and a quota number k, based on the HDM, CDH utilizes the community characteristics and modifies the degree centrality for efficiently selecting the initial k "influential" individuals in order to maximize the number of cascade adoptions by which these individuals will influence others.

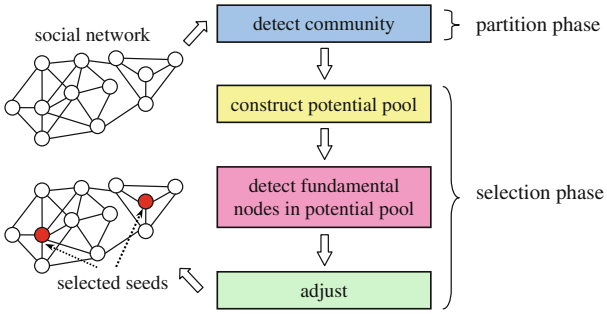


Fig. 1 Overview of the community and degree heuristic approach

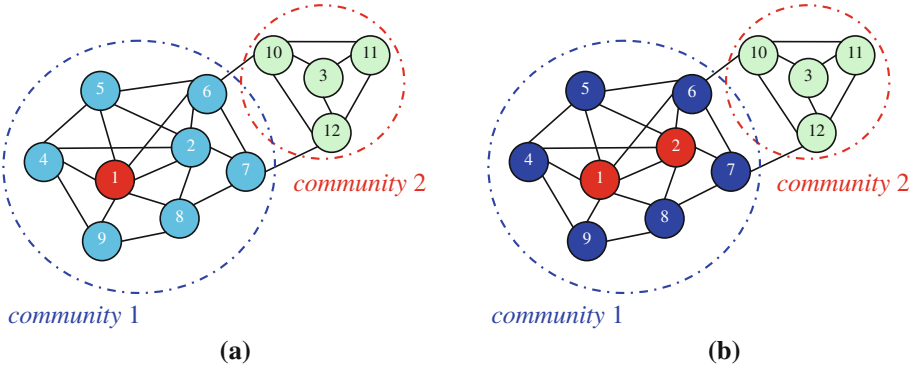


Fig. 2 **a** The distribution of heat where node 1 is the seed. **b** The distribution of heat where node 1 and node 2 are the seeds

Initially, we select k individuals as seeds, denoted by the set $S = \{s_1, s_2, \ldots, s_k\}$, and the k seeds are given a certain amount of heat h_0, and then we find the influence spreads, that is, the number of influenced nodes, based on the HDM with G and S. At time zero of the heat diffusion process, we set $f_i(t_0) = h_0$ for $v_i \in S$. As time elapses, the heat will diffuse throughout the whole social network. If the amount of heat of individual vi at time t is greater than or equal to an activation threshold θ, this individual v_i will be considered as having been successfully influenced or activated by others. We define the set of influenced nodes of S, denoted as $I_s(t)$, to be the expected number of individuals who will adopt the product at time t. Now, the *Influence Maximization Problem* could be interpreted as: finding the most influential k-size set S to maximize the size of set $I_s(t)$ at time t, where $I_s(t) = \{i | f_i(t) \geq \theta, i \leq N\}$. This problem is NP-hard, as already proven in [17]. We select the HDM to be our diffusion model since it can realistically simulate a real-world social network.

The proposed CDH is composed of two phases, the partition phase and the selection phase. The partition phase detects the communities of the social network, where a community is a subset of individuals who interact with each other more frequently than other individuals outside the community. Based on the communities discovered, in the selection phase, we propose mechanisms to select seed nodes. Explicitly, in real life, one's information often spreads around one's circle of friends. In other words, most of one's influence clearly spreads within one's own community. We can also find the same phenomenon in the HDM. Figure 1 illustrates the framework of the CDH. For example, in Fig. 2, the color of each node signifies

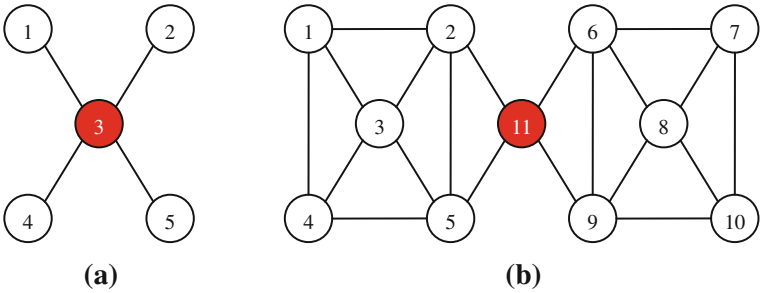


Fig. 3 Two examples of fundamental nodes. **a** Node 3 is a fundamental node. **b** Node 11 is a fundamental node

the amount of heat, that is, more dark blue means a larger amount of heat. Nodes in the same community are captured by the dotted circle. As in Fig. 2a, if we choose node 1 as the seed, we can see that most gains of heat are the nodes in node 1's community. However, if we choose node 1 and node 2 as the seeds, most heat is also spread around node 1's community, as in Fig. 2b. Nodes in the other community gain very little heat. We can conclude that if we choose many nodes in the same community as seeds, most gains of heat are in their own community, and the nodes in other communities gain little heat. Therefore, information about the community is very useful to avoid the influence of overlapping in the HDM.

The reason for using community detection algorithms rather than a conventional graph partitioning algorithm is that we want to detect "the best," or in other words, the "most natural" partitioning of a network without providing any heuristic information, such as the number of partitions. For example, if it is natural to partition the network into 3 communities, then we should not force the network to be partitioned into 4 communities. The second phase, the *selection phase*, finds the most influential nodes based on the result of the *partition phase* and some parameters of the HDM, that is, flow duration, thermal conductivity and activation threshold. For example, in Fig. 2a, community 1 is larger than community 2. We can observe that selecting nodes in community 1 as seeds instead of nodes in community 2 could trigger more individuals to be activated. The degree of each node in a social network also fits the power-law distribution [1,2], that is, a very large number of nodes have a very small number of neighbors. Hence, most large-degree nodes are in large communities. Due to this reason, we only consider nodes in the large communities as seed candidates and put them in a potential pool.

Then, we detect the "fundamental nodes" from the potential pool. A fundamental node means a node that has more potential to be a seed due to (1) a larger degree than other nodes in the same community or (2) location at an important position in the network. An important position means one that connects many communities. Figure 3a, b shows two kinds of fundamental nodes. As in Fig. 3a, node 3 is a fundamental node since it has the largest degree among all nodes. In Fig. 3b, node 11 is a fundamental node since it has a better position that can easily influence two sets {1, 2, 3, 4, 5} and {6, 7, 8, 9, 10}. How to detect the fundamental nodes is one of the differences between the CDH-Kcut and CDH-SHRINK algorithms.

Although fundamental nodes have a good chance of becoming final seeds, they are not the best seeds in different situations (parameters) of the HDM. For example, seeds that perform well in short flow duration may not be good in long flow duration. Therefore, adjusting the fundamental nodes to become more significant seeds is very important.

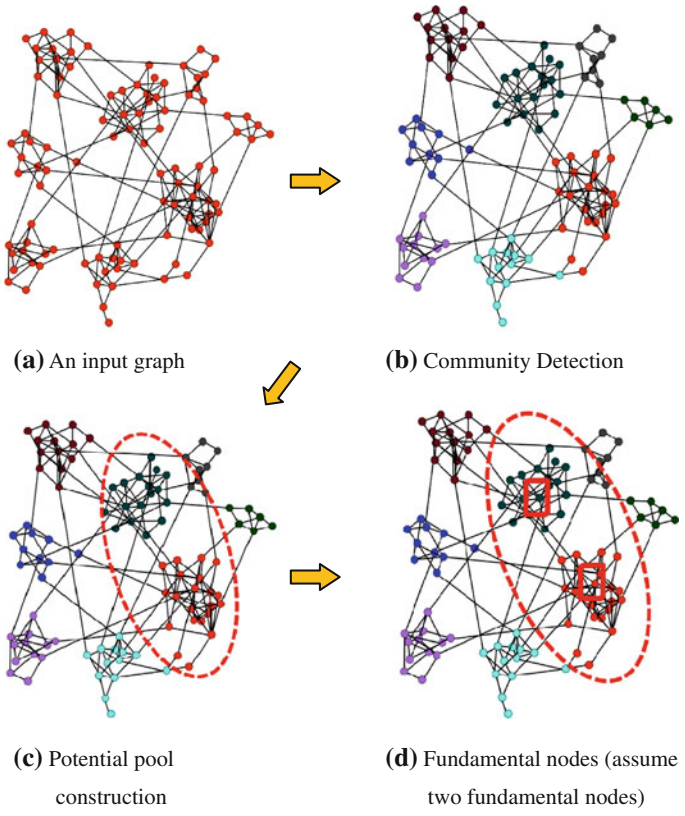


Fig. 4 An example of the concept of the community and degree heuristic approach

We use an example, as shown in Fig. 4, to describe the concept of the CDH. Given a social network as in Fig. 4a, we first detect the communities of the network. Figure 4b is the result of community detection where different colors mean different communities. After the partition phase, the first step of the selection phase is to construct the potential pool. The two communities circled by red dotted circles are potential pools since they are the two largest communities among all communities, as shown in Fig. 4c. Assume two seeds are needed for a user. Figure 4d shows that two fundamental nodes are selected in the potential pool since they have the largest degree. After the step of constructing the potential pool and finding the fundamental nodes, we effectively reduce the number of candidates for seed selection. Based on the concept of CDH, two algorithms, CDH-Kcut and CDH-SHRINK, are proposed to efficiently solve the maximum influence problem.

3.2 CDH-Kcut

CDH-Kcut utilizes the Kcut algorithm to first discover communities in the partition phase. Note that the Kcut algorithm [21] combines the recursive partitioning and direct k-way method based on the eigenvectors of the Laplacian matrix of a graph. It achieves the efficiency of a recursive approach, while also having the same accuracy as a direct k-way method. If there are multiple communities, using multiple eigenvectors to directly compute a k-way partition

Algorithm 1: *CDH-Kcut* (G, k, p)

Input: Graph of social network G ; number of total seeds k ; parameters p
Output: k seeds

01: **call** *Kcut* (G);
02: $S \leftarrow \varnothing$; // seed set
03: select top-k biggest communities from the communities in *Kcut* (G);
04: **for each** selected community SC_i **do**
05: add top-p degree nodes into set SC_i^p ;
06: **for each** SC_i^p **do**
07: $S \leftarrow S \cup$ the most degree node in SC_i^p ;
08: $I_s(t) \leftarrow$ execute HDM on G with S; //$I_s(t)$: the set of influenced nodes of current S
09: $IM \leftarrow |I_s(t)|$; // IM : save the max number of influenced nodes
10: **for each** community SC_i **do**
11: **if** size(SC_i) > avg($\sum_{i=1}^{k}$size(SC_i)) **then**
12: add SC_i in LC; //LC : the set of large communities
13: sort LC based on community size;
14: $di \leftarrow 0$; // di: index of *d_node*
15: **for each** SC_i in LC
16: $ai \leftarrow 2$; //ai: index of *a_node*
17: **while** *true* **do**
18: *a_node* $\leftarrow$ the *ai-th* large degree node in SC_i^p ;
19: *d_node* $\leftarrow$ the seed candidates $s_{k\text{-}di}$ in S ;
20: replace the *d_node* with *a_node* in S ;
21: $I_s(t) \leftarrow$ execute HDM on G with S;
22: **if** $|I_s(t)| < IM$ **then**
23: restore the replacement in line 20;
24: **break**;
25: $IM \leftarrow |I_s(t)|$;
26: $ai \leftarrow ai + 1$; $di \leftarrow di + 1$;
27: output S;

Fig. 5 Pseudo code of the CDH-Kcut algorithm

is better than using the recursive bi-partitioning method. Every node in social network graph G will belong to only one community, and overlapping communities are not allowed in Kcut. We assume that G is partitioned into l communities. In most cases, l is larger than k, so in this paper, we do not discuss the case $l < k$. For more details of the Kcut algorithm, interested readers could refer to [21].

Figure 5 illustrates the pseudo code of the CDH-Kcut algorithm. First, we select the top-k biggest communities from the discovered communities by Kcut, denoted as SC_i, $1 \leq i \leq k$ (lines 1–2, algorithm 1). Then, we construct a potential pool, $PP(G)$ (lines 3–5, algorithm 1). A potential pool is defined as, $PP(G) = \{SC_1^p, SC_2^p, \ldots, SC_k^p\}$, where SC_i^p is the set of top-p degree nodes in the i-th largest community SC_i. $PP(G)$ keeps the top-p degree nodes in each community of the top-k largest communities. In most cases, $p = 10\,\%$ of community size is enough for selecting good seeds. Then, the fundamental nodes are selected from the potential pool. Since Kcut cannot identify the important location of nodes in each SC_i^p, degree is considered as the only attribute that distinguishes good fundamental nodes. We select the

largest degree node in each SC_i^p as the set of fundamental nodes $S = \{s_1, s_2, \ldots, s_k\}$ (lines 6–8, algorithm 1). By potential pool and fundamental node construction, we can significantly narrow down the range of possible seeds (Fig. 5).

Moreover, we adjust the fundamental nodes to be the final seeds. Our basic idea of adjustment is a heuristic that tries to use an add-node, *a_node*, to replace a delete-node, *d_node*. By our observation, seeds selected from a large community could trigger more adoptions of a product or an innovation than seeds selected from a small community. For example, in Fig. 3a, community 1 is large and community 2 is small. If the size of a community is larger than $AvgSC = \text{avg}(\Sigma_{i=1}^{k}\text{size}(SC_i))$, this community is deemed as being large (lines 10–13, algorithm 1). As a result, we are inclined to select an *a_node* from large communities and a *d_node* from small communities. We would like to investigate whether selecting nodes from large communities can gain more influence spread (lines 17–20, algorithm 1). If the number of influenced nodes after replacing is larger than before replacing, we replace the *d_node* with an *a_node* in S (lines 21–24, algorithm 1). Notice that delete-nodes must be fundamental nodes, that is, the *d_node* is selected from S. Finally, we output the nodes in set S as the selected seeds (line 27, algorithm 1).

Moreover, we adjust the fundamental nodes to be the final seeds. Our basic idea of adjustment is a heuristic that tries to use an add-node, *a_node*, to replace a delete-node, *d_node*. By our observation, seeds selected from a large community could trigger more adoptions of a product or an innovation than seeds selected from a small community. For example, in Fig. 3a, community 1 is large and community 2 is small. If the size of a community is larger than $AvgSC = \text{avg}(\Sigma_{i=1}^{k}\text{size}(SC_i))$, this community is deemed as a large community (lines 10–13, algorithm 1). As a result, we are inclined to select an add-node from large communities and a delete-node from small communities. We would like to investigate whether selecting nodes from large communities can gain more influence spread (lines 17–20, algorithm 1). If the number of influenced nodes after replacing is larger than before replacing, we replace the delete-node, *d_node* with an add-node, *a_node* in S (lines 21–24, algorithm 1). Notice that delete-nodes must be fundamental nodes, that is, the *d_node* is selected from S. Finally, we output the nodes in set S as the selected seeds (line 27, algorithm 1).

The adjustment is to avoid the influence spread being spoiled by the effect of different parameter values, such as flow duration, activation threshold and thermal conductivity. We discuss these effects individually. Comparing the different effects by time duration, information will diffuse farther in long flow duration. That is, in long flow duration, the seeds would influence more individuals than in short flow duration. Therefore, we should not select too many seeds from a single community in long flow duration. In contrast, it is appropriate to select more seeds from a single community if the flow duration is very short.

It is more difficult to make individuals adopt products if the activation threshold is high. Individuals need more heat to be activated with a higher activation threshold, so we tend to select more seeds in a community with a high activation threshold. High thermal conductivity makes information diffuse more quickly. Compared with low thermal conductivity, information with high thermal conductivity diffuses over a longer distance. Hence, we do not select many seeds from a community with high thermal conductivity. We conclude that different parameters may cause different levels of seed clustering. It is better to select more seeds in one community, that is, to have a higher level of seed clustering with short flow duration, a high threshold and low thermal conductivity. On the contrary, in long flow duration, low threshold, and high thermal conductivity social networks, not many seeds in a single community are needed, that is, there is a lower level of seed clustering. Therefore, the adjustment in CDH-Kcut is to test and verify whether large communities should need more seeds.

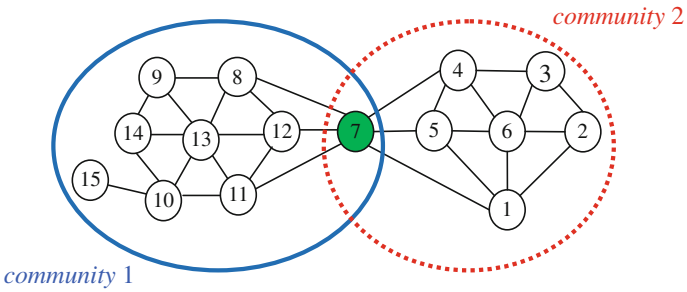


Fig. 6 Illustration of community size reduction

3.3 CDH-SHRINK

As with CDH-Kcut, CDH-SHRINK also consists of two phases, the partition phase and the selection phase. However, SHRINK [13] is utilized to discover communities in the partition phase. SHRINK is a parameter-free hierarchical network clustering algorithm combining the advantages of density-based clustering and modularity optimization methods. It uses the density-based method to quickly discover which set of nodes may be in the same cluster and then uses modularity optimization to decide whether the results of the clustering are good or not. SHRINK not only detects hierarchical communities but also identifies hubs and outliers. For community partition, a *hub*, that is, a node connecting different communities, can provide more information about the community structure properties. As in the example in Fig. 6, node 7 is a hub connecting communities 1 and 2. A brief introduction to the SHRINK algorithm is provided here; please refer to [13] for more details. Since the communities detected by SHRINK are more precise than those found by Kcut, we can select more productive fundamental nodes.

In the selection phase, we construct the potential pool and select fundamental nodes and then adjust the fundamental nodes to find final seeds. Since the hub can provide more information about community structure, constructing the potential pool and selecting fundamental nodes are quite different from the same process in CDH-Kcut. If we have to find k seeds, we need k iterations of selecting the fundamental nodes. In the $i-$th iteration, $1 \leq i \leq k$, we only choose the largest community SC_i among all remaining communities and select the top-p degree nodes in SC_i, denoted as SC_i^p, for the potential pool construction, and then select a fundamental node from SC_i^p. Based on the observation of experiments, $p = 10\,\%$ of SC_i's size is sufficient to select good seed nodes. After selecting a fundamental node in SC_i^p (how selection is made is discussed later), the size of each un-chosen community covered by this fundamental node has to reduce its degree. For example, as shown in Fig. 6, assume that node 7 is the fundamental node in community 1 (size $= 10$) in the 1st iteration. In the 2nd iteration, when we choose the biggest community, the size of community 2 (size $= 7$) will reduce the degree of node 7 in community 2, that is, $7 - 3 = 4$. Community size reduction can effectively avoid influence overlap.

Now, we discuss how to select a fundamental node in the potential pool. A good fundamental node should "cover" communities as much as possible while having as much influence on its communities as possible. In Fig. 7, node$z$ belongs to communities C_1, C_2 and C_3. That is, z covers C_1, C_2 and C_3. We define two evaluation metrics, *position score* and *hub purity*, to measure how good a fundamental node is.

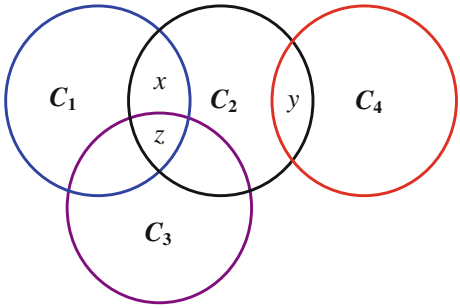


Fig. 7 An example of how to compute purity. C_1, C_2, C_3, and C_4 are communities

Definition 1 *(Position Score and Hub Purity)* To evaluate the importance of a node's position in a network, the *position score* of a node u is defined as the number of communities which u belongs to, that is, $position\_score(u) = |\{C_i | u \in C_i, u \in V$ and $C_i \in$ set of discovered communities$\}|$. If the node u is a non-hub node, the *position_score*(u) is 1. Otherwise, the *position_score*(u) is larger than 1. The *hub purity* of a node is defined as, $hub\_purity(u) = \frac{|\{C_i | u \in C_i, u \text{ is a } hub \text{ and } C_i \notin FC\}|}{position\_score(u)}$, where FC is the set of communities, which contains fundamental node u and u is a hub. For example, in Fig. 7, C_1 and C_2 are communities containing fundamental node x. C_1, C_2 and C_3 are communities containing node z. C_2 and C_4 are communities containing node$y$. Therefore, *position_score*(z) = 3, *hub_purity* (z) = 1/3 and *position_score*(y) = 2, *hub_purity* (y) = 1/2.

We choose the "MAX priority" nodes from SC_i^p as fundamental nodes, $1 \leq i \leq k$. Selecting fundamental nodes in CDH-SHRINK is different from the same process in CDH-Kcut. To compare the *priority* of nodes, we use the function *compare_priority*, which works as follows,

1) If both nodes are hubs, we compare their *position_score* other than their degree. We compare hubs according to how many communities they belong to, since we want to cover more communities. That is, we want to choose a hub that has important positions in the network. Besides, if the node is a hub, its purity must exceed the purity threshold. A low-purity hub may cause information overlap since it may cover too many communities.
2) When comparing non-hub nodes with either hub or non-hub nodes, we compare their influence on their neighbors, that is, degree, due to the non-availability of information about the importance of the location of the non-hub nodes.

After comparing the top-p degree nodes in SC_i, we can successfully find the fundamental nodes. The fundamental node may have a very good position that connects SC_i with many other communities and has much influence on its neighbors. The set of selected fundamental nodes is denoted as $S = \{s_1, s_2, \ldots, s_k\}$, $1 \leq i \leq k$.

Finally, we adjust the fundamental nodes to be the final seed nodes. Adjustment in CDH-SHRINK is also a heuristic, which tries to choose an add-node to replace a delete-node. Then, we test whether the influence spread (influenced nodes) after replacing is larger than before replacing. Two evaluation metrics, *left* and *seed load*, are defined as the auxiliaries to determine the add-nodes and delete-nodes.

Definition 2 *(Left and Seed Load)* Given a community $C_i \in$, a set of discovered communities in G, we define $left(C_i) = |\{u | u \in C_i$ and u is a non-activated node$\}|$. *left* (C_i) might be thought of as "the need to add more seeds to C_i." As *left* (C_i) increases, the need to select more seeds for C_i increases. We define $seed\_load(C_i) = \frac{\text{size}(C_i)}{|\{u | u \in C_i, u \in S\}|}$. The implication of

Algorithm 2: *CDH-SHRINK* (G, k, p)

Input: Graph of social network G ; number of total seeds k ; parameters p
Output: k seeds
01: **call** *SHRINK* (G);
02: $S \leftarrow \emptyset$; $SC \leftarrow \emptyset$; // seed set
03: **while** $|SC| < k$ **do** // select the fundamental node
04: $SC \leftarrow SC \cup$ biggest community SC_i among all remaining communities;
05: add top-p degree nodes of SC_i into set SC_i^p ;
06: **for each** node in SC_i^p **do**
07: max_node = compare_priority (n_i, max_node); // n_i : the i-th largest degree node in SC_i^p
08: $S \leftarrow S \cup max\_node$;
09: **for each** community C_i which has max_node **do**
10: $size(C_i) = size(C_i) - degree(max\_node)$;
11: $I_s(t) \leftarrow$ execute HDM on G with S; //$I_s(t)$: the set of influenced nodes of current S
12: $IM \leftarrow |I_s(t)|$; // IM : save the max number of influenced nodes
13: **for** 1 to r **do** //adjustment
14: $a\_node \leftarrow u \mid max\{\Sigma_{i=1}^{t}\, left\,(C_i \mid u \in C_i), t = position\_score\,(u)\}$;
15: $d\_comm \leftarrow \min_{Ci \in SC}\, seed\_load\,(C_i)$;
16: $d\_node \leftarrow \min_{u \in S,\, u \in d\_comm} |\{v \mid v \in A(u), A(u) \subseteq G(V)\}|$;
17: replace the d_node with a_node in S;
18: $I_s(t) \leftarrow$ execute HDM on G with S;
19: **if** $I_s(t) < IM$ **then**
20: restore the replacement in line 17;
21: $IM \leftarrow I_s(t)$;
22: Output individuals in S;

Procedure: *Compare_priority* (a, b)
23: **if** (a is hub) $\wedge$ (hub_purity (a) < $purity\_threshold$) **then**
24: return b;
25: **if** (a is hub) $\wedge$ (b is hub) **then**
26: return max ($position\_score$ (a), $position\_score$ (b));
27: **if** a is non-hub **then**
28: return max ($degree(a)$, $degree(b)$);

Fig. 8 Pseudo code of the CDH-SHRINK algorithm

seed_load (C_i) is that there are too many seeds in C_i. When *seed_load* (C_i) is small, that perhaps means that there are too many seeds in C_i.

In each iteration, we firstly select the add-node, $a\_node = u \mid \max\{\Sigma_{i=1}^{t}\, left(C_i \mid u \in C_i), t = position\_score(u)\}$ and choose a delete-community, d_comm, which has the smallest *seed_load* among SC_i, $1 \leq i \leq k$. We define a function $A(u)$ that outputs a set of active nodes adjacent to node u. Subsequently, we select the delete-node, d_node, which has the minimum $|A(d\_node)|$ among all seeds in d_comm. Finally, we test whether we should substitute an add-node for the delete-node. If the influence spread after substitution is more than before, we make a substitution. To quickly find a productive a_node, we do not consider very low degree nodes. We could assume that selecting low degree nodes as seeds is not productive.

The pseudo code of the CDH-SHRINK algorithm is given in Fig. 8. The information of community structure and hubs can be discovered by the SHRINK algorithm (line 1,

Table 1 Parameters of LFR benchmark graphs [15,16]

Parameters	Description
N	Number of nodes
M	Number of edges
maxd	Maximum degree
mp	Mixing parameter (each node shares a fraction *mp* of its edges with nodes in other communities)

Table 2 Five generated synthetic networks

Dataset	N	M	*maxd*	*mp*
1000*Smp*	1,000	9,097	100	0.1
1000*Lmp*	1,000	9,097	100	0.5
1000*Lmaxd*	1,000	9,097	200	0.1
1000*LM*	1,000	22,484	100	0.1
5000*Smp*	5,000	47,094	100	0.1

algorithm 2). If k seeds are desired, k iterations of selecting fundamental nodes (lines 3–9, algorithm 2) are executed. Then, we adjust the fundamental nodes to select the final seeds (lines 13–21, algorithm 2). The adjustment uses r iterations to test the substitution. In most cases, $r = 2k \sim 3k$ is sufficient to obtain satisfactory influence spread. Finally, we output the nodes in set S as the selected seeds (line 22, algorithm 2).

4 Experiments

To evaluate the performance of the CDH, three influence maximization algorithms, CDH-Kcut, CDH-SHRINK, and the enhanced greedy algorithm (EGA) [17], are implemented for comparison. We also implement a naïve algorithm, degree heuristic (DH) as a baseline, which only selects the top-k largest degree nodes as the seed nodes. All algorithms are designed based on the HDM, are implemented in the C++ language, and are tested on a Pentium D 3.0 GHz with 2 GB of main memory running the Windows XP system. The comprehensive performance study is conducted on five synthetic networks and three real-world datasets, the karate network [30], the NETHep network, and the Facebook network. In each experiment, we vary three parameters, the activation threshold (θ), flow duration (t), and thermal conductivity (α) of the HDM, to compare the influence spread (number of activated nodes) and efficiency (execution time) of the four algorithms.

4.1 Synthetic networks

The synthetic datasets in the experiments are generated using a synthetic generation program, Lancichinetti–Fortunato–Radicchi (LFR) benchmark graphs [15,16]. The parameter setting of the LFR generator is shown in Table 1. We generate five different undirected graphs, (1) 1000*Smp*: the graph with 1,000 nodes and small mixing parameter; (2) 1000*Lmp*: the graph with 1,000 nodes and large mixing parameter; (3) 1000*Lmaxd*: the graph with 1,000 nodes and large maximum degree; (4) 1000*LM*: the graph with 1,000 nodes and large number of degree; and (5) 5000*Smp*: the graph with 5,000 nodes and small mixing parameter, as shown in Table 2. Generally, the higher the mixing parameter of a network, the more difficult it is to reveal the community structure.

Table 3 The influence spread of different algorithms in 1000*Smp* (1,000 nodes and small mixing parameter, the largest influence spread is highlighted in boldface)

Parameter setting	DH	EGA	CDH-SHRINK	CDH-Kcut
$t = 0.1, \theta = 0.1, \alpha = 0.1$	215	**341**	**341**	339
$t = 0.1, \theta = 0.2, \alpha = 0.1$	215	**336**	335	332
$t = 0.1, \theta = 1.5, \alpha = 0.1$	95	**223**	221	216
$t = 0.1, \theta = 2.0, \alpha = 0.1$	95	133	141	**166**
$t = 0.2, \theta = 0.1, \alpha = 0.1$	336	**386**	379	345
$t = 0.3, \theta = 0.1, \alpha = 0.1$	472	503	499	**508**
$t = 0.4, \theta = 0.1, \alpha = 0.1$	567	635	628	**649**
$t = 0.1, \theta = 0.1, \alpha = 0.2$	336	**386**	**386**	345
$t = 0.1, \theta = 0.1, \alpha = 0.3$	472	**503**	498	**503**
$t = 0.1, \theta = 0.1, \alpha = 0.4$	567	635	632	**649**

Table 4 The influence spread of different algorithms in 1000*Lmp* (1,000 nodes and large mixing parameter, the largest influence spread is highlighted in boldface)

Parameter setting	DH	EGA	CDH-SHRINK	CDH-Kcut
$t = 0.1, \theta = 0.1, \alpha = 0.1$	206	**251**	249	233
$t = 0.1, \theta = 0.2, \alpha = 0.1$	187	**225**	**225**	215
$t = 0.1, \theta = 1.5, \alpha = 0.1$	105	**176**	170	149
$t = 0.1, \theta = 1.6, \alpha = 0.1$	56	**137**	131	114
$t = 0.2, \theta = 0.1, \alpha = 0.1$	484	**562**	559	535
$t = 0.3, \theta = 0.1, \alpha = 0.1$	722	**790**	782	763
$t = 0.4, \theta = 0.1, \alpha = 0.1$	829	**892**	888	853
$t = 0.1, \theta = 0.1, \alpha = 0.2$	484	**562**	560	520
$t = 0.1, \theta = 0.1, \alpha = 0.3$	722	**790**	781	742
$t = 0.1, \theta = 0.1, \alpha = 0.4$	829	**892**	885	863

Table 3 provides the results of the four algorithms with different activation thresholds (θ), flow duration (t), and thermal conductivity (α) in 1000*Smp*. The influence spreads of CDH-SHRINK and CDH-Kcut from $\theta = 0.2$ to $\theta = 1.4$ are almost the same. Hence, we only discuss $\theta = 0.2$, $\theta = 1.5$ and $\theta = 2.0$. We can observe that CDH-SHRINK and CDH-Kcut have the same influence spread in most cases and are even better than EGA with $\theta = 2.0$. Figure 9a, b shows 5 seed nodes selected by CDH-Kcut with t= 0.1, θ = 0.1, α = 0.1 and with t= 0.1, θ = 0.2, α = 0.1 in 1000*Smp*, respectively. We can see that a higher activation threshold leads to the phenomenon of seed clustering.

Table 4 shows the influence spread of different algorithms with different parameter settings in 1000*Lmp*. CDH-Kcut performs worse than CDH-SHRINK in 1000*Lmp* since SHRINK could detect a more accurate community structure than Kcut. Hence, if it is difficult to identify the correct community structure of the network, CDH-SHRINK will probably perform better than CDH-Kcut. The accuracy of the detected community structure reflects the performance of the influence spread of CDH-SHRINK and CDH-Kcut. Therefore, with increased mixing parameters, that is, each node shares a fraction of its edges with nodes in other communities, the influence spread performance of CDH-SHRINK and CDH-Kcut deteriorates. Table 5 indicates the influence spread of different algorithms with different activation thresholds, flow duration, and thermal conductivity in 1000*Lmaxd*. Nodes in 1000*Lmaxd* could have

Table 5 The influence spread of different algorithms in 1000*Lmaxd* (1,000 nodes and large maximum degree, the largest influence spread is highlighted in boldface)

Parameter setting	DH	EGA	CDH-SHRINK	CDH-Kcut
$t = 0.1, \theta = 0.1, \alpha = 0.1$	202	332	**333**	315
$t = 0.1, \theta = 0.2, \alpha = 0.1$	196	**290**	288	269
$t = 0.1, \theta = 1.5, \alpha = 0.1$	146	**170**	161	143
$t = 0.1, \theta = 1.6, \alpha = 0.1$	136	**138**	120	113
$t = 0.2, \theta = 0.1, \alpha = 0.1$	299	494	**495**	473
$t = 0.3, \theta = 0.1, \alpha = 0.1$	404	565	**569**	561
$t = 0.4, \theta = 0.1, \alpha = 0.1$	482	**627**	620	599
$t = 0.1, \theta = 0.1, \alpha = 0.2$	299	494	**495**	473
$t = 0.1, \theta = 0.1, \alpha = 0.3$	404	565	**569**	561
$t = 0.1, \theta = 0.1, \alpha = 0.4$	482	**627**	620	599

Table 6 The influence spread of different algorithms in 1000*LM* (1,000 nodes and large number of edge, the largest influence spread is highlighted in boldface)

Parameter setting	DH	EGA	CDH-SHRINK	CDH-Kcut
$t = 0.1, \theta = 0.1, \alpha = 0.1$	521	**663**	653	653
$t = 0.1, \theta = 0.2, \alpha = 0.1$	316	**506**	499	487
$t = 0.1, \theta = 1.5, \alpha = 0.1$	246	**430**	**430**	412
$t = 0.2, \theta = 0.1, \alpha = 0.1$	184	**226**	**226**	218
$t = 0.3, \theta = 0.1, \alpha = 0.1$	816	**923**	919	898
$t = 0.4, \theta = 0.1, \alpha = 0.1$	939	**991**	989	976
$t = 0.1, \theta = 0.1, \alpha = 0.2$	996	**1,000**	993	**1,000**
$t = 0.1, \theta = 0.1, \alpha = 0.3$	816	**923**	919	898
$t = 0.1, \theta = 0.1, \alpha = 0.4$	939	**991**	989	976

larger degrees. That is, the degree of some nodes will be much larger than that of others. Consequently, the performance of the influence spread of DH will be improved, especially with a high activation threshold.

Table 6 shows the influence spread of the four algorithms with different parameters in 1000*LM*. In 1000*LM*, each node has more neighbors, so information will spread quickly. Hence, we could see that the influence spread in 1000*LM* is higher than that in 1000*Smp*, 1000*Lmp*, and 1000*Lmaxd*. In most cases, the influence spreads of CDH-SHRINK and CDH-Kcut are still better than that of DH. Table 7 indicates the influence spread of different algorithms with different activation thresholds, flow duration and thermal conductivity in 5000*Smp*. 5000*Smp* has 5,000 nodes. In most cases, the influence spread of CDH-SHRINK and CDH-Kcut is almost equal to that of EGA and still better than DH.

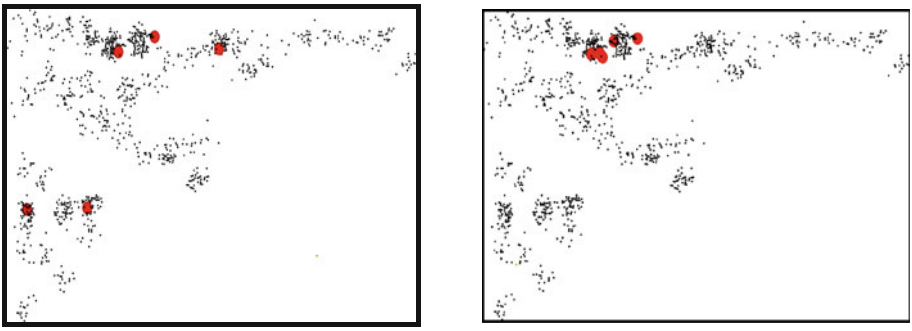
In summary, CDH-SHRINK, in general, is more productive than CDH-Kcut. With the increase in the activation threshold, seeds will cluster together to trigger a larger influence spread. As shown in Fig. 9a, b, the graphs indicate the seed clustering phenomenon with a high activation threshold.

4.2 Zachary's karate network

Zachary's karate network [30] consists of 34 nodes and 78 edges. Nodes represent the members of a karate club in the United States who have been observed over a period of

Table 7 The influence spread of different algorithms in 5000*Smp* (5,000 nodes and small mixing parameter, the largest influence spread is highlighted in boldface)

Parameter setting	DH	EGA	CDH-SHRINK	CDH-Kcut
$t = 0.1, \theta = 0.1, \alpha = 0.1$	391	**540**	**540**	536
$t = 0.1, \theta = 0.2, \alpha = 0.1$	252	**438**	433	426
$t = 0.1, \theta = 1.5, \alpha = 0.1$	210	**397**	393	356
$t = 0.2, \theta = 0.1, \alpha = 0.1$	169	**261**	255	232
$t = 0.3, \theta = 0.1, \alpha = 0.1$	843	**1,258**	1, 238	1,202
$t = 0.4, \theta = 0.1, \alpha = 0.1$	877	**1,284**	1, 275	1,213
$t = 0.1, \theta = 0.1, \alpha = 0.2$	952	**1,451**	1, 423	1,378
$t = 0.1, \theta = 0.1, \alpha = 0.3$	843	**1,258**	1, 238	1,202
$t = 0.1, \theta = 0.1, \alpha = 0.4$	877	**1,284**	1, 275	1,213



(a) $t = 0.1, \theta = 0.1, \alpha = 0.1$ **(b)** $t = 0.1, \theta = 0.2, \alpha = 0.1$

Fig. 9 5 Seeds (*red nodes*) selected by CDH-Kcut in 1000*Smp* (color figure online)

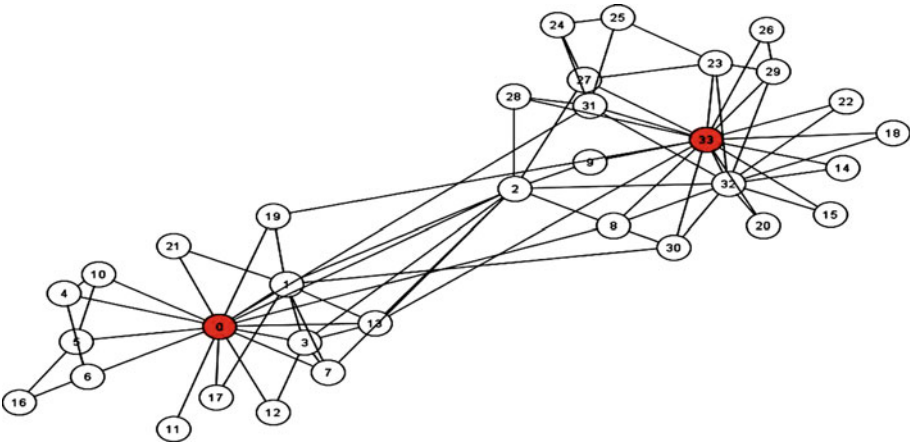


Fig. 10 Zachary's karate network. *Red nodes* are node 0 and node 33 (color figure online)

3 years. Edges connect individuals who have been observed to interact outside the activities of the club. The Zachary's karate network is shown in Fig. 10. Table 8 lists the final seed selection (the two numbers in parentheses) and the influence spread of the four algorithms

Table 8 The influence spread and two selected seeds of different algorithms in Zachary's karate network (the largest influence spread is highlighted in boldface)

Parameter setting	DH	EGA	CDH-SHRINK	CDH-Kcut
t= 0.1, θ= 0.1, α= 0.1	**31 (0, 33)**	**31 (0, 33)**	**31 (0, 33)**	**31 (0, 33)**
t= 0.1, θ= 0.2, α= 0.1	6 (0, 33)	**12 (32,33)**	**12 (32, 33)**	**12 (32, 33)**
t= 0.1, θ= 0.3, α= 0.1	6 (0, 33)	**12 (32, 33)**	**12 (32, 33)**	**12 (32, 33)**
t= 0.1, θ= 0.2, α= 0.2	**31 (0, 33)**	**31 (0, 33)**	**31 (0, 33)**	**31 (0, 33)**
t= 0.4, θ= 0.6, α= 0.1	6 (0, 33)	8 (4, 7)	**12 (32, 33)**	**12 (32, 33)**

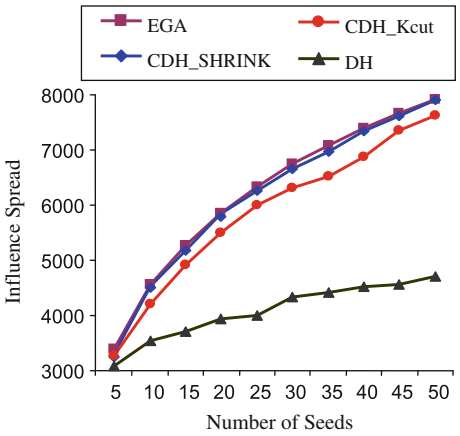


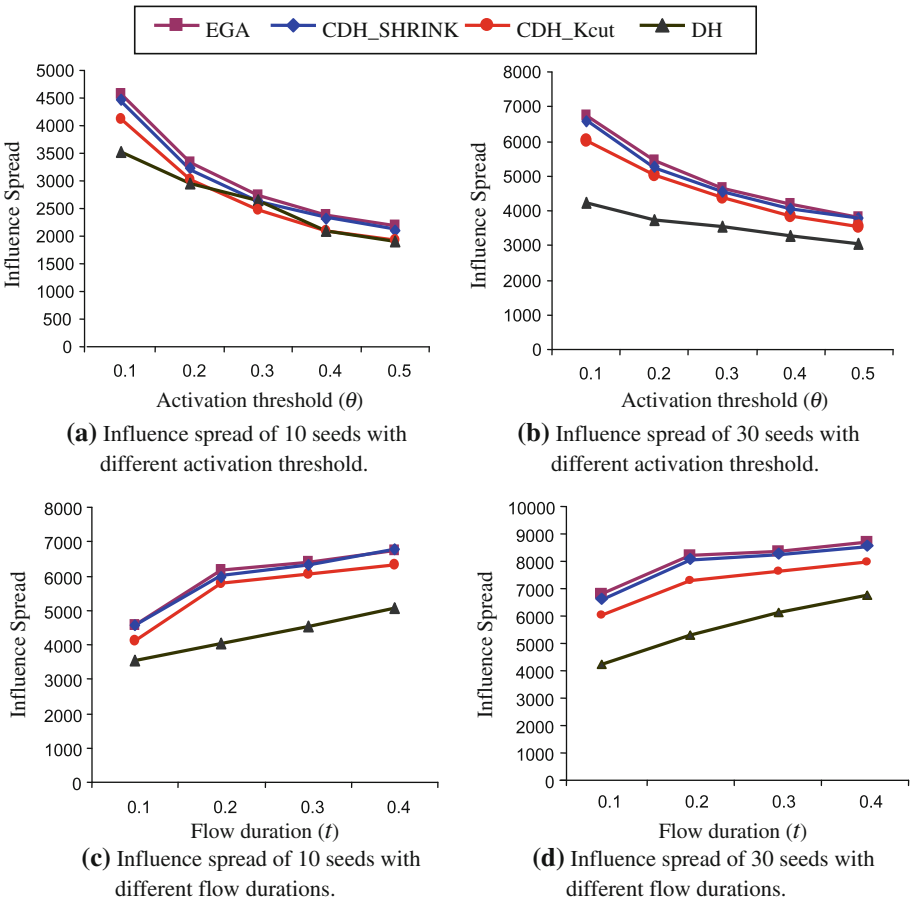
Fig. 11 Influence spread of different algorithms on NETHep with $t = 0.1, \theta = 0.1, \alpha = 0.1$

with different parameter settings. From Table 8, we can find that CDH-SHRINK and CDH-Kcut could select good seeds according to different parameter values, but DH could not. Consequently, CDH-SHRINK and CDH-Kcut get the same influence spread as EGA in most cases. In case t= 0.1, θ = 0.2 and α = 0.1, two seeds, 0 and 33, are selected, as the red nodes in Fig. 10. Furthermore, in some cases, like t = 0.4, θ =0 .6, α = 0.1, the CDH strategy gets better influence spread. We could see that the two seeds selected with t= 0.1, θ = 0.2 and α = 0.1 are 32 and 33. That is, a high activation threshold easily causes the phenomenon of seed clustering while high thermal conductivity does not. Therefore, two seeds 0 and 33 are selected with t= 0.1, θ = 0.2, and α = 0.2 that are the same seeds as those with t= 0.1, θ = 0.1, and α = 0.1.

4.3 NETHep network

In this section, we extract a large real-life academic collaboration network, NETHep from the e-print. Each node in the network represents an author. If an author i co-authored a paper with author j, the graph contains an undirected edge from i to j. If the paper is co-authored by k authors, this generates a completely connected graph on k nodes. Including all papers from the period January 1993 to April 2003 (124 months), NETHep contains 12,008 nodes and 237,010 edges.

On the large real collaboration network, NETHep, we report the efficiency and influence spread of EGA, CDH-SHRINK, CDH-Kcut, and DH with different numbers of seeds and values of parameters. In CDH-SHRINK, the purity threshold is set to 0.35, which can get satisfactory influence spread. Figure 11 shows the influence spread of different algorithms with different numbers of seeds on NETHep. The x-axis indicates the number of seeds



(a) Influence spread of 10 seeds with different activation threshold.

(b) Influence spread of 30 seeds with different activation threshold.

(c) Influence spread of 10 seeds with different flow durations.

(d) Influence spread of 30 seeds with different flow durations.

Fig. 12 Influence spread of different algorithms on NETHep with different parameter settings

and the y-axis indicates influence spread. In most cases, EGA's influence spread $\approx$ CDH-SHRINK's influence spread $>$ CDH-Kcut's influence spread $>$ DH's influence spread. With the increasing number of seeds, CDH-SHRINK and CDH-Kcut improve and are better than DH since most seeds selected by DH are only in a few communities.

Figure 12a, b shows the influence spread of 10 seeds and 30 seeds with different θ from 0.1 to 0.5 with a span of 0.1, respectively. The x-axis indicates the activation threshold, and the y-axis indicates the influence spread. The results reflected in the figures show that although the total influence spread of the four algorithms will decrease as θ increases, CDH-SHRINK and CDH-Kcut still maintain a good influence spread. Notice that DH improves its influence spread with the increase of θ, which results from the effect of seeds with high θ. However, it is still worse than CDH-SHRINK and CDH-Kcut when selecting more seeds. Figure 12c, d indicates the influence spread of 10 seeds and 30 seeds with different t from 0.1 to 0.4 with a span of 0.1, respectively. The x-axis describes flow duration, and the y-axis describes influence spread. The figures show that our proposed algorithms still maintain good influence spread with the increase of t. We only report results from $t = 0.1$ to $t = 0.4$ since a t value that is too large will lead to the situation that most nodes are influenced, and thus, we cannot easily distinguish the performance of the four algorithms.

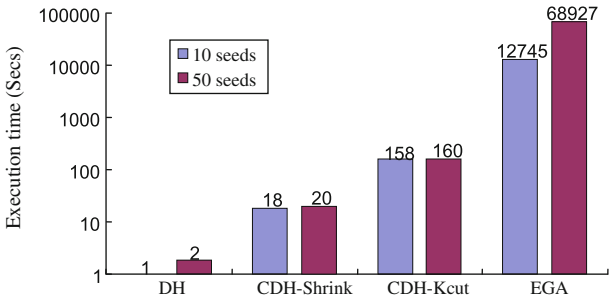


Fig. 13 Running time of different algorithms on the NETHep network when selecting 10 seeds and 50 seeds, respectively

We present the execution times of the four algorithms in Fig. 13 with 10 seeds and 50 seeds. The x-axis indicates different algorithms, and the y-axis (logarithmic scale) indicates the execution time. Since DH only needs to select the top-k degree nodes as seeds, its execution time is extremely efficient. CDH-SHRINK has the shortest execution time among the other three algorithms and is about 3,446 times faster than EGA (68927/20). We can also see that the running time of EGA is proportional to the number of seeds. The execution times of CDH-SHRINK and CDH-Kcut are only slightly different between 10 seeds and 50 seeds. This is because CDH-SHRINK and CDH-Kcut only have to spend a little more time on adjustment when the number of nodes increases. As shown is Figs. 11 and 13, although on average EGA is about 1.3 % better than CDH-SHRINK in terms of influence spread, the execution time is much slower than the proposed CDH methods. Nowadays, social networks are becoming bigger and bigger. If a time-consuming algorithm takes a long time to decide which set of individuals should be seed nodes, its selection may be ineffective and it will lose superiority due to the dynamic variation of the networks. The efficiency of an algorithm is also a critical issue.

4.4 Facebook network

In this section, we discuss the influence maximization on a large real-life dataset, the Facebook network. Each node in the network represents a user. If user i is a friend of user j, the graph contains an undirected edge from i to j. The network is denoted as FB, in the period from April 2004 to 2009 January (124 months), and contains 63,731 nodes and 817,090 edges. We not only analyze the efficiency but also the influence spread of our algorithms with respect to different numbers of seeds and parameter values. In CDH-SHRINK, the purity threshold is set to 0.2, which can demonstrate satisfactory influence spread in FB.

Figure 14 shows the influence spread of different algorithms with different numbers of seeds on FB. The x-axis indicates the number of seeds, and the y-axis indicates the influence spread. As shown in Fig. 14, in most cases, EGA's influence spread $\approx$ CDH-SHRINK's influence spread $>$ CDH-Kcut's influence spread $>$ DH's influence spread. Since EGA is too time-consuming, we only report the influence spread from 5 seeds to 30 seeds.

Figure 15a, b depicts the influence spread of 10 seeds and 30 seeds with different θ from 0.1 to 0.4 with a span of 0.1, respectively. The x-axis indicates the activation threshold, and the y-axis indicates the influence spread. We can observe that, when only 10 seeds are selected, EGA's influence spread $\approx$ CDH-SHRINK's influence spread $>$ CDH-Kcut's influence spread $>$ DH's influence spread, with $\theta = 0.2$, 0.3 and 0.4. In Fig. 15b, we can

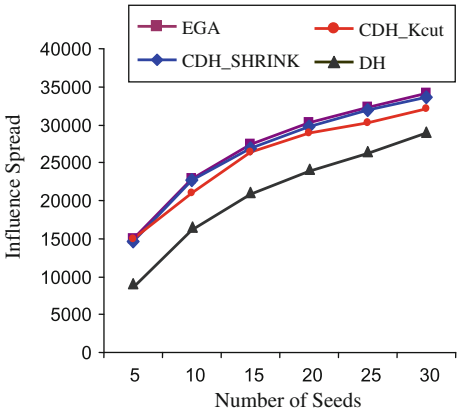
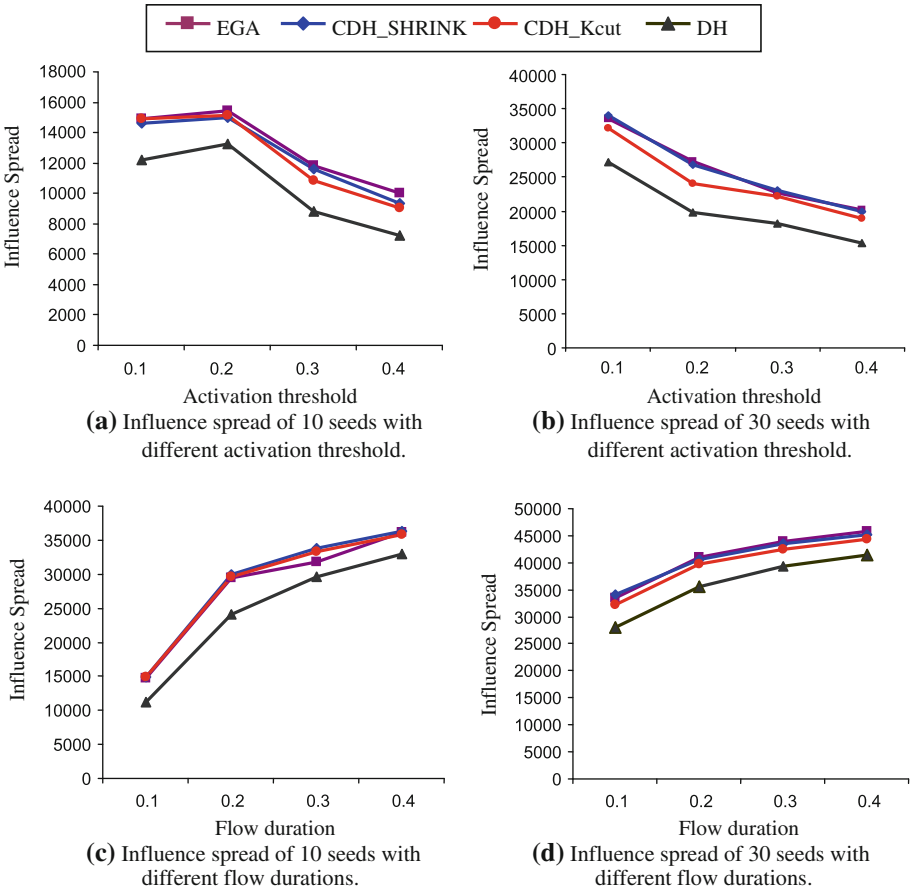


Fig. 14 Influence spread of different algorithms on FB with $t = 0.1, \theta = 0.1, \alpha = 0.1$



(a) Influence spread of 10 seeds with different activation threshold.

(b) Influence spread of 30 seeds with different activation threshold.

(c) Influence spread of 10 seeds with different flow durations.

(d) Influence spread of 30 seeds with different flow durations.

Fig. 15 Influence spread of different algorithms on FB with different parameter settings

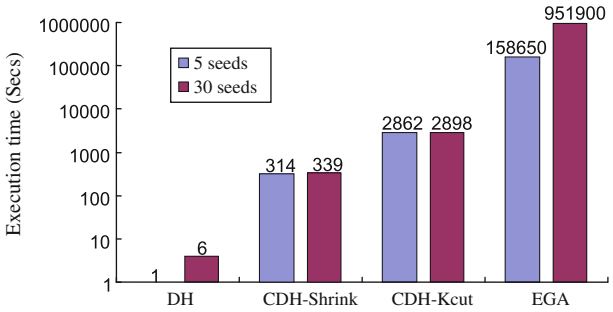


Fig. 16 Execution time of the different algorithms on the FB network when selecting 5 seeds and 30 seeds

see that DH's influence spread is better than that in Fig. 15a. Moreover, CDH-SHRINK's influence spread is better than that of EGA when $\theta = 0.1$ and 0.3.

Figure 15c, d illustrates the influence spread of 10 seeds and 30 seeds with different t from 0.1 to 0.4 with a span of 0.1, respectively. The x-axis indicates the flow duration, and the y-axis indicates the influence spread. Unlike other cases on NETHep or FB, in Fig. 15c, CDH-SHRINK's influence spread > CDH-Kcut's influence spread > EGA's influence spread > DH's influence spread. However, when we select 30 seeds as shown in Fig. 15d, the ranking of influence spread becomes CDH-SHRINK $\approx$ EGA > CDH-Kcut > DH. EGA still has the most influence spread in most cases. It is noticed that no matter what parameter values are set on FB, CDH-SHRINK's influence spreads are very close to those of EGA.

We present the execution times of the four algorithms in Fig. 16 with 5 seeds and 30 seeds, respectively. The x-axis indicates the different algorithms, and the y-axis (logarithmic scale) indicates the execution time. CDH-SHRINK has the shortest running time. Just like on NETHep, we can see that the running time of EGA is proportional to the number of seeds. Overall, in terms of influence spread, CDH-SHRINK $\approx$ EGA > CDH-Kcut > DH; in terms of efficiency, DH > CDH-SHRINK > CDH-Kcut $\gg$ EGA. One point that deserves mentioning is that due to the phenomenon of seed clustering, DH will get better performance in terms of influence spread with high activation than with a low activation threshold.

5 Conclusion and future work

In this paper, we present two algorithms, CDH-Kcut and CDH-SHRINK, adopting the undirected HDM by integrating the information of community structure and the modified degree-centrality method. The purpose of our work is to solve the influence maximization problem efficiently and still have good influence spread based on the HDM. The experimental results on the real-world and synthetic datasets also validate that our proposed algorithms achieve great performance in running time and influence spread. When comparing CDH-SHRINK with CDH-Kcut, CDH-SHRINK, in general, utilizes hubs and better community structure to achieve better influence spread. Although both algorithms are efficient with time complexity $O(M \log N)$, CDH-Kcut, in practice, would take more execution time.

In the future, to interpret the real world more realistically, the influence maximization problem can be extended to weighted graphs. Furthermore, dynamic evolution is also an important property of social networks. Static community detection algorithms could only detect community structure without considering the evolution of social networks. It would

be a worthwhile investigation to utilize dynamic community structures to solve the influence maximization problem.

Acknowledgments Suh-Yin Lee was supported by the National Science Council, Project No. NSC99-2221-E-009-128-MY2. Wen-Chih Peng was supported in part by the National Science Council, Project No. 100-2218-E-009-016-MY3 and 100-2218-E-009-013-MY3, by Taiwan MoE ATU Program, by ITRI JRC, Project No. B352BW3300, by D-Link and by Microsoft.

References

1. Albert R, Jeong H, Barabasi A (1999) Diameter of the World Wide Web. Nature 401:130–131
2. Barabasi A, Albert R (1999) Emergence of scaling in random networks. Science 286:509–512
3. Bortner D, Han J (2010) Progressive clustering of networks using structure—connected order of traversal. In: 26th IEEE international conference on data, engineering (ICDE'10), pp 653–656
4. Chen W, Wang Y, Yang S (2009) Efficient influence maximization in social networks. In: Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining (KDD'09), pp 199–208
5. Danon L, Duch J, Diaz-Guilera A, Arenas A (2005) Comparing community structure identification. J Stat Mech Theory Exp 09:P09008
6. Domingos P (2005) Mining social networks for viral marketing. IEEE Intell Syst 20(1):80–93
7. Domingos P, Richardson M (2001) Mining the network value of customers. In: Proceedings of the 7th ACM SIGKDD international conference on knowledge discovery and data mining (KDD'01), pp 57–66
8. Domingos P, Richardson M (2002) Mining knowledge-sharing sites for viral marketing. In: Proceedings of the 8th ACM SIGKDD international conference on knowledge discovery and data mining (KDD'02), pp 61–70
9. Ester M, Kriegel H, Sander J, Xu X (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of 2nd international conference on knowledge discovery and data Mining (KDD'96), pp 226–231
10. Estevez P, Vera P, Saito K (2007) Selecting the most influential nodes in social network. In: Proceedings of the international joint conference on, neural networks (IJCNN'07), pp 2397–2402
11. Feng Z, Xu X, Yuruk N, Schweiger T (2007) A novel similarity-based modularity function for graph partitioning. In: Proceedings of the 9th international conference on data warehousing and knowledge, discovery (DaWaK'07), pp 385–396
12. Goldenberg J, Libai B, Muller E (2001) Talk of network: a complex systems look at the underlying process of word-of-mouth. Mark Lett 12(3):211–223
13. Huang J, Sun H, Han J, Deng H, Sun Y, Liu Y (2010) SHRINK: a structural clustering algorithm for detecting hierarchical communities in networks. In: Proceedings of the 19th ACM conference on information and, knowledge management (CIKM'10), pp 219–228
14. Kempe D, Kleinberg J, Tardos E (2003) Maximizing the spread of influence through a social network. In: Proceedings of the 9th ACM SIGKDD international conference on knowledge discovery and data mining (KDD'03), pp 137–146
15. Lancichinetti A, Fortnato S, Kertesz J (2009) Detecting the overlapping and hierarchical community structure in complex network. New J Phys 11(3):033015
16. Lancichinetti A, Fortnato S, Radicchi F (2008) Benchmark graphs for testing community detection algorithms. Phys Rev E 78(4):046110
17. Ma H, Yang H, Lyu M, King I (2008) Mining social networks using heat diffusion processes for marketing candidates selection. In: Proceedings of the 17th ACM conference on information and, knowledge management (CIKM'08), pp 233–242
18. Ng A, Jordan M, Weiss Y (2001) On spectral clustering: analysis and an algorithm. In: Neural information processing systems: natural and synthetic (NIPS 2001), pp 849–856
19. Palla G, Derenyi I, Farkas I, Vicsek T (2005) Uncovering the overlapping community structure of complex networks in nature and society. Nature 435:814–818
20. Rogers E (2003) Diffusion of innovations. Free Press, New York
21. Ruan J, Zhang W (2007) An efficient spectral algorithm for network community discovery and its applications to biological and social networks. In: Proceedings of the 7th IEEE international conference on data mining (ICDM'07), pp 643–648
22. Saito K, Kimura M, Ohara K, Motoda H (2011) Efficient discovery of influential nodes for SIS models in social networks. Knowl Inf Syst 30(3):613–635

23. Valente T (1995) Network models of the diffusion of innovations. Hampton Press, Cresskill
24. Wan L, Liao J, Zhu X (2008) Finding and evaluating community structure in social networks. In: Proceedings of the 4th international conference on advanced data mining and applications (ADMA'08), pp 620–627
25. Wang Y, Feng X (2009) A potential-based node selection strategy for influence maximization in a social network. In: Proceedings of the 5th international conference on advanced data mining and applications (ADMA'09), pp 350–361
26. Wasserman S, Wasserman S, Faust K (1994) Social network analysis: methods and applications. Cambridge University Press, Cambridge
27. White S, Smyth P (2005) A spectral clustering approach to finding communities in graph. In: Proceedings of the 5th SIAM international conference on data mining (SDM'05), pp 274–286
28. Xu X, Yuruk N, Feng Z, Schweiger T (2007) SCAN: a structural clustering algorithm for networks. In: Proceedings of the 13th ACM SIGKDD international conference on knowledge discovery and data mining (KDD'07), pp 824–833
29. Young H (2000) The diffusion of innovations in social networks. The Johns Hopkins University, economics working paper 437
30. Zachary W (1997) An information flow model for conflict and fission in small group. J Anthropol Res 33:452–473

Author Biographies

Yi-Cheng Chen received the B.S. degree in computer science from Yuan Ze University, Taiwan, in 2000, and the M.S. degree in computer science from National Taiwan University of Science and Technology, Taiwan, in 2002, and the Ph.D. degree in computer science from National Chiao Tung University. He is an assistant researcher in the department of Computer Science, National Chiao Tung University, Taiwan. His research interests include sequential pattern mining, cloud computing, social network analysis, bioinformatics and data mining.

Wen-Chih Peng was born in Hsinchu, Taiwan, R.O.C in 1973. He received the BS and MS degrees from the National Chiao Tung University, Taiwan, in 1995 and 1997, respectively, and the Ph.D. degree in Electrical Engineering from the National Taiwan University, Taiwan, R.O.C in 2001. Currently, he is an associate professor at the department of Computer Science, National Chiao Tung University, Taiwan. Prior to joining the department of Computer Science and Information Engineering, National Chiao Tung University, he was mainly involved in the projects related to mobile computing, data broadcasting and network data management. Dr. Peng published some papers in several prestigious conferences, such as IEEE International Conference on Data Engineering (ICDE), IEEE International Conference on Data Mining (ICDM) and ACM Conference on Information and Knowledge Management (ACM CIKM) and prestigious journals (e.g., IEEE TKDE, IEEE TMC, IEEE TPDS). Dr. Peng has the best paper award in ACM Workshop on location-based social network 2009 and the best student paper award in IEEE International Conference on Mobile Data Management 2011. His research interests include mobile computing, network data management and data mining. He is a member of IEEE.

Suh-Yin Lee received the B.S. degree in electrical engineering from National Chiao Tung University, Taiwan, in 1972, and the M.S. degree in computer science from University of Washington, U.S.A., in 1975, and the Ph.D. degree in computer science from Institute of Electronics, National Chiao Tung University. She has been a professor in the Department of Computer Science and Information Engineering at National Chiao Tung University since 1991 and was the chair of that department in 1991–1993. Her research interests include content-based indexing and retrieval, distributed multimedia information system, mobile computing, and data mining.