CIM: Community-Based Influence Maximization in Social Networks

YI-CHENG CHEN, Tamkang University
WEN-YUAN ZHU and WEN-CHIH PENG, National Chiao Tung University
WANG-CHIEN LEE, Pennsylvania State University
SUH-YIN LEE, National Chiao Tung University

Given a social graph, the problem of influence maximization is to determine a set of nodes that maximizes the spread of influences. While some recent research has studied the problem of influence maximization, these works are generally too time consuming for practical use in a large-scale social network. In this article, we develop a new framework, *community-based influence maximization* (CIM), to tackle the influence maximization problem with an emphasis on the time efficiency issue. Our proposed framework, CIM, comprises three phases: (i) community detection, (ii) candidate generation, and (iii) seed selection. Specifically, phase (i) discovers the community structure of the network; phase (ii) uses the information of communities to narrow down the possible seed candidates; and phase (iii) finalizes the seed nodes from the candidate set. By exploiting the properties of the community structures, we are able to avoid overlapped information and thus efficiently select the number of seeds to maximize information spreads. The experimental results on both synthetic and real datasets show that the proposed CIM algorithm significantly outperforms the state-of-the-art algorithms in terms of efficiency and scalability, with almost no compromise of effectiveness.

Categories and Subject Descriptors: J.4 [**Computer Applications**]: Social and Behavioral Sciences; H.2.8 [**Data Management**]: Database Applications—*Data Mining*

General Terms: Algorithms, Theory, Measurement

Additional Key Words and Phrases: Community detection, diffusion models, influence maximization, social network analysis

ACM Reference Format:
Yi-Cheng Chen, Wen-Yuan Zhu, Wen-Chih Peng, Wang-Chien Lee, and Suh-Yin Lee. 2014. CIM: Community-based influence maximization in social networks. ACM Trans. Intell. Syst. Technol. 5, 2, Article 25 (April 2014), 31 pages.
DOI: http://dx.doi.org/10.1145/2532549

1. INTRODUCTION

Owing to the advances in Web 2.0 technologies and ideas, various online social networking services have emerged. They allow users to establish social connections (i.e., friendship) and facilitate interactions and share thoughts, comments, pictures, etc., among friends. The friendship among users naturally forms a social network, which becomes a valuable marketing media, as the size of user bases in social networking services rapidly grows over the years. As a result, many companies have increasingly chosen online social networks over other media (such as newspaper and television) for marketing campaigns. According to a report from eMarketer (`http://www.emarketer.com/`),

Authors' addresses: Yi-Cheng Chen (corresponding author), Department of Computer Science and Information Engineering, Tamkang University; email: ejen.cs95g@nctu.edu.tw; Wen-Yuan Zhu, Wen-Chih Peng, and Suh-Yin Lee, Department of Computer Science, National Chiao Tung University; Wang-Chien Lee, Department of Computer Science and Engineering, Pennsylvania State University.

DOI: http://dx.doi.org/10.1145/2532549

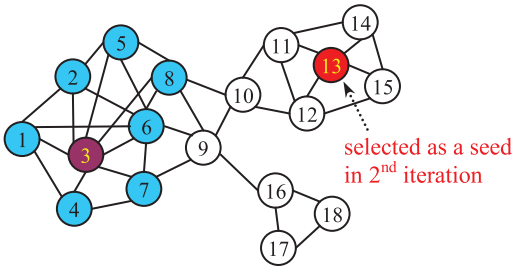


Fig. 1. An illustration of the enhanced greedy algorithm for the influence maximization problem.

advertisement spending on worldwide social network sites has obtained a steady double-digit growth for the past several years.

Finding influential individuals is one of the most critical issues for marketing on social-networking media. For example, to plan for a marketing campaign of a new product via social-networking media, a company may want to target a small number of users (referred to as *seeds*) for a trial of the product, hoping that these users could influence their friends, who may in turn influence their friends, to buy the product (or become familiar with it). The idea is that through the effect of *word-of-mouth*, the company could reach a large number of potential customers. This scenario is thus formally defined as the influence maximization problem [Domingos and Richaidson 2001], which aims to select the initial seeds who may influence a maximal number of users to adopt a marketed product.[1]

A considerable amount of research effort has been put forth on the influence maximization problem [10, 12, 23, 29]. Kempe et al.'s seminal work [2003] proves that the influence maximization problem is NP-hard and proposes Monte Carlo simulation-based algorithms to solve the problem. Nevertheless, as social networks become gigantic, the *efficiency* of mining algorithms for influence maximization becomes very critical (e.g., the timeliness required for a product campaign may be lost if it takes weeks to select a set of seeds for promotion of the products). Although some approximate algorithms (e.g., [Chen et al. 2009; Estevez et al. 2007]) have been proposed to address the efficiency issue, these works are mostly dependent on diffusion models that do not capture the temporal factor of diffusion (such as linear threshold model or independent propagation model [Estevez et al. 2007; Goldenberg et al. 2001; Valente 1995; Young 2000]). It is quite important to capture the temporal process of information diffusion for realistic market planning, for example, some marketing campaigns may want to time the chain effect of product adoption in three days, a week, or a month. Thus, in this article, we adopt the Heat Diffusion Model (HDM) [Ma et al. 2008] due to its strength in capturing the temporal diffusion process.

The influence maximization problem based on HDM is also NP-hard [Ma et al. 2008]. While a heuristic algorithm, called *enhanced greedy algorithm* (EGA) [Ma et al. 2008], has been proposed to select influential nodes greedily, it still incurs excessive computation. Basically, EGA iteratively selects a node as a seed until k seeds are obtained. At each iteration, it will combine each non-seed node, one by one, with the seed nodes selected from previous iterations to compute their combined influence spreads. The node introducing the most influence spread to the existing seeds is selected as a new seed. Consider the example social network in Figure 1. Suppose that node 3 has been selected as a seed in the first iteration. To find the next seed, EGA computes the influence spreads of (3, 1), (3, 2), (3, 4), . . . , (3, 17) and eventually selects node 13 as

[1]Note that the number of those influenced users is referred to as influence spread in the literature.

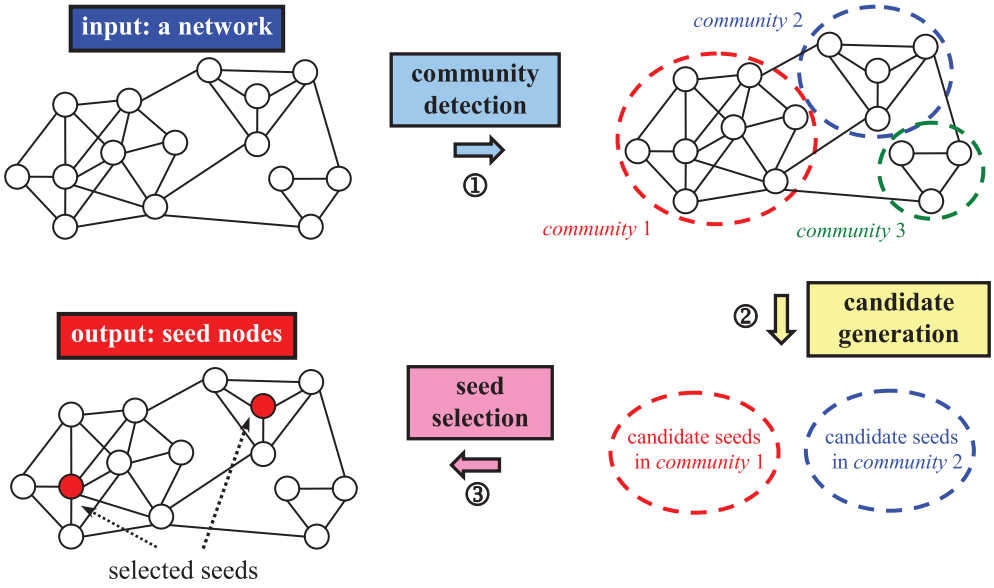


Fig. 2. Overview of the CIM framework approach.

the next seed, because it results in the most increase in influence spread. Next, to find the third seed, EGA continues the aforementioned process to compute influence spread for (3, 13, 1), (3, 13, 2), (3, 13, 4), . . . , (3, 13, 12), (3, 13, 14), . . . (3, 13, 17). Obviously, this greedy method is very time consuming. Based on our observation, computation of influence spread associated with some non-seed nodes is unnecessary in the second iteration. As shown in Figure 1, suppose node 3 has been selected as a seed; nodes 1, 2, 4, 5, 6, 7, and 8 actually share many common neighbor nodes with node 3. If we choose one of them as a seed, the information will only spread among those common neighbor nodes. On the other hand, if we select node 13 as an additional seed, the increase of spread is more significant. Hence, if we can identify nodes that share a lot of common neighbors, we may be able to save significant computational overhead. Notice that the nodes that share a lot of common neighbors are known as a *community structure* in the literature of social network analysis and data mining. Thus, we argue, by exploring the community structures naturally embedded in a social network, efficient algorithms can be developed to address the influence maximization problem.

In this article, we develop a new framework, namely, *community-based influence maximization* (CIM), to tackle the problem of the influence maximization problem, with an emphasis on the time efficiency issue.[2] Figure 2 provides an overview of our approach in CIM, which comprises three phases: (i) community detection, (ii) candidate generation, and (iii) seed selection. Specifically, phase (i) discovers the community structure of the network; phase (ii) uses the information of communities to narrow down the possible seed candidates; and phase (iii) finalizes the seed nodes from the candidate set. Several issues arise in CIM. (1) As already mentioned, community structure provides a means to identify nodes with overlapping influence spreads and thus reduces redundant computation of influence spreads. Does any community obtained by an arbitrary clustering algorithm work well in CIM? How shall we design a clustering algorithm to find "good" clusters? Most clustering algorithms typically come with a

[2]Preliminary result of this work has been reported in SNAKDD'12 [Chen et al. 2012].

number of parameters which are hard for users to tune. Could we minimize the parameter setting? (2) As social networks in realistic settings are extremely large, the search space of selecting seeds is also huge. Effectively reducing the number of candidate seeds is a critical issue of influence maximization. How to narrow down the size of a candidate set of seeds is a core issue in CIM. (3) The computation of influence spreads is most time consuming. Thus, how to reduce the overhead incurred for following the diffusion model is a major bottleneck and thus critical to CIM.

The contributions of this article are as follows.

—We observe that nodes in social networks naturally cluster together. This finding prompts us to develop a new framework for the influence maximization problem, namely, community-based influence maximization (CIM), which explores the community information to effectively reduce computation of overlapped influence spreads to achieve high efficiency.
—Aiming to obtain good clusters for CIM, we develop a hierarchical clustering algorithm, called *H_Clustering*, to detect the community structure of a social graph. We find that the size of the community and the ratio between the required seeds and nodes of the graph are effective for filtering out insignificant nodes. Hence, we adopt these ideas for candidate reduction in CIM.
—We argue that a large community may have a large potential influence spread, and thus we use the total number of nodes in the community to decide the number of seeds to be allocated. This idea avoids the greedy selection of seed nodes and significantly reduces the overhead for running the HDM model.
—We conduct a comprehensive evaluation by experimentation using real datasets. The experimental results show that algorithm CIM significantly outperforms the state-of-the-art algorithms in terms of both efficiency and influence spread.

The rest of this article is organized as follows. Section 2 describes the research problem, discusses our observations, and reviews some relevant works. Section 3 details the CIM framework and associated algorithms. Section 4 presents the experiments on several synthetic and real datasets. Finally, Section 5 concludes.

2. PRELIMINARIES

In this section, we first formulate the influence maximization problem based on the heat diffusion model, then we discuss some observations that lead to our strategies in addressing the targeted research problem, and finally we review some related works.

2.1. Problem Formulation

In this article, a social network is modeled as an undirected graph $G(V, E)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is the vertex set and $E = \{(v_i, v_j) \mid v_i, v_j \in E\}$ is the set of edges in the graph. A node represents an individual, and an edge between two nodes represents their social relationship (e.g., friendship or co-authorship, to name a few). 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 influence maximization is given here.

Influence Maximization Problem. Given a social network $G = (V, E)$ and a number k, the task is to determine k seeds (i.e., nodes) such that these seeds could spread their influence to other nodes with an objective of maximizing the number of nodes influenced by the seeds.

As mentioned earlier, in order to capture the temporal process of information diffusion, which is important for market planning, we consider the heat diffusion model (HDM) [Ma et al. 2008] to model the information diffusion on social networks. HDM,

formally described next, is a realistic model that simulates social behavior in accordance with a physical phenomenon, *heat flow (diffusion)*.

Heat Diffusion Model (HDM). Let $f_i(t)$ denote the heat at node v_i at time t. Suppose that an initial distribution of heat starts at time zero, and at time t, each node v_i receives an amount of heat from its neighbor v_j during a period Δt. The heat received is 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

$$\frac{f_i(t + \Delta t) - f_i(t)}{\Delta t} = \alpha \Sigma_{j:(vj,vi)\in E}(f_j(t) - f_i(t)) = \alpha H f(t), \tag{1}$$

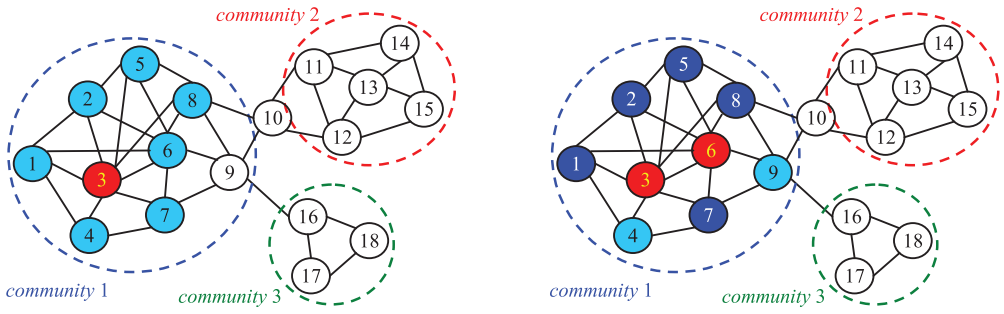
where H is a matrix and α is the heat diffusion coefficient. If the amount of heat of node v exceeds the activation threshold θ, we think node v will purchase a product or adopt an innovation.

Influence Maximization Problem Based on HDM. Given a social network G based on HDM, by selecting k individuals in V as seeds denoted as a set $S = \{s_1, s_2, \ldots, s_k\}$, and giving each of them a certain amount of heat h_0, we can obtain the influenced spreads (i.e., the number of influenced nodes) with G and S, as follows. At time t_0 of the heat diffusion process, we set $f_{si}(t_0) = h_0$ for each $s_i \in S$. As time elapses, the heat will diffuse throughout G based on Eq. (1). If the amount of heat of each individual $v_i \in V$ at time t is larger than or equal to an activation threshold θ, v_i will be considered as being successfully activated (or influenced). We define the set of active nodes influenced by S at time t as $I_s(t)$. Accordingly, the influence maximization problem based on HDM is interpreted as follows: given a social network G (V, E), find a set S (consisting of k seeds) to maximize the size of active nodes $I_s(t)$ at time t, where $I_s(t) = \{v_i \mid f_{vi}(t) \geq \theta, v_i \in V\}$.

2.2. Observations

In this article, we aim to tackle the efficiency issue of the influence maximization problem based on HDM by exploring the community structures embedded naturally in social networks. Based on our preliminary analysis of the influence maximization problem and the properties of community structures, we made the following observations which raise some potential issues and provide good guidance for algorithm designs in the CIM framework. First, we argue that nodes in social networks naturally cluster together. To effectively reduce the overhead incurred in computing influence spreads based on HDM, we shall explore the clustering phenomenon among nodes in a community to avoid the computation of overlapped influence spreads among nodes in the same community. Consider Figure 3(a) as an example where we can observe three communities in the graph. Suppose that we want to select two seeds. If we choose node 3 as a seed and give it an amount of heat (i.e., influence), it will spread to nodes 1, 2, 4, 5, 6, 7 and 8. Suppose we choose node 6 as another seed; most of the heat will flow to the nodes that have already been activated, as shown in Figure 3(b). The darker color means the node has a greater amount of heat. We can conclude that if we choose many nodes in the same community as seeds, most gains in heat are within their own community instead of other communities.

Second, in reality, the social network is usually extremely huge. How to effectively reduce the search space of seed nodes is an important issue. By analyzing the community structures, we observe that not all communities are significant enough to accommodate seed nodes. For example, in Figure 3, although we have three communities in the network, community 3 may be too small to be worth placing a seed there, because selecting the nodes in community 3 as a seed may only activate three nodes initially.



(a) The distibution of information spread, where node 3 is the seed.

(b) The distibution of information spread, where node 3 and node 6 are the seeds.

Fig. 3. The community structure of an example network.

Hence, it is insignificant in comparison with other large communities. We may prune off insignificant communities and their nodes from consideration, thereby reducing the search space of seed selection.

Third, while we pointed out earlier that we may want to avoid placing multiple seeds in the same community, the size of community is still a factor for seed placement. We should not treat every community the same, as placing seeds in a large community could trigger more adoptions of a product or an innovation than in a small community. For example, in Figure 3, if we want to select three seeds, we should select two seeds in community 1 and one seed in community 2 instead of selecting one seed in each community. Finally, nodes connecting many communities (called *hubs*), for example, node 10 in Figure 3, may play an important role in influence maximization problem, as the influence can spread easily from hubs to many different communities.

2.3. Related Works

In this section, we review some closely related existing works, categorized as (1) diffusion models, (2) influence maximization algorithms, and (3) community clustering algorithms.

Diffusion Models. Diffusion is a type of communication concerning the spread of messages perceived as new ideas or innovations. Rogers [2003] theorizes the diffusion process where an innovation is communicated via certain channels among the members of a society. Due to emerging online social networking services and their potential in viral marketing, diffusion models in social networks have received growing research interests in recent years [Estevez et al. 2007; Goldenberg et al. 2001; Ma et al. 2008; Valente 1995; Young 2000]. In addition to the heat diffusion model (HDM) adopted in our research work, here we review some other well-known models.

Consider an undirected social graph G (V, E). Let $N(v) = \{u \mid (u, v) \in E\}$ be the neighbor set of node v and b_{uv} be the influence of active node u on its inactive neighbor v. The linear threshold model (LTM) defines $A(v)$ as the set of active nodes in $N(v)$, $(A(v) \subseteq N(v))$, denoting those who have adopted an innovation or a marketed product. Given an activation threshold θ, for a node v, if $\Sigma_{u \subseteq A(v)}\, b_{uv} \geq \theta$, node v becomes active. The intuition is that, for an inactive node v, if the total influence exerted by all its active neighbors exceeds θ, node v becomes active. LTM assumes that the newly activated node v will also exert influence on its inactive neighbors and thus make some inactive neighbors become active. This process will continue until no node can be activated. On the other hand, as its name suggests, the independent cascade model (ICM) assumes

that a node u, once activated, will try to activate its inactive neighbor v with a success probability p, independent of the influence of other neighbor nodes. Notice that ICM assumes that an active node u has only one chance to activate its neighbor v. If it fails, it will not be able to activate v anymore.

Time plays an essential role in diffusion flows in a social network. Some models focus on inferring the underlying dynamics over a network [Gomez-Rodriguez et al. 2010]. NETINF [Gomez-Rodriguez et al. 2010] investigates the problem of tracing paths of diffusion and infers network connectivity using submodular optimization. CONNIE [Myers et al. 2010] infers the connectivity and a prior probability of influence for every edge using a convex program and some heuristics. However, both models force the transmission rate between all nodes to be fixed. By allowing transmission at different rates across different edges, NETRATE [Gomez-Rodriguez et al. 2011] is able to infer temporally heterogeneous interactions within a network. Thus, NETRATE can simulate the temporal dynamics over the underlying network.

Compared to the HDM adopted in our study, the aforementioned ICM and LTM are not useful for predicting the future behavior of the network [Domingos 2005; Domingos and Richardson 2002]. Some other diffusion models [Estevez et al. 2007; Young 2000] have been proposed recently, but they are mostly variations of LTM and ICM. On the contrary, NETINF, CONNIE, and NETRATE focus on inferring the tracing path of diffusion and temporal dynamics over a given network [Gomez-Rodriguez et al. 2010, 2011; Myers et al. 2010]. A social network is a very complex network with all kinds of messages flowing within it. Modeling social network marketing realistically is extremely difficult. As pointed out in Ma et al. [2008], HDM provides more realistic parameters to simulate the conditions of diffusion in the real world, such as time and thermal conductivity. Thus, we adopt HDM as the diffusion model in this work.

Influence Maximization Algorithms. The influence maximization problem, under the context of various diffusion models, has been shown to be NP-hard in general. Many works have been proposed to obtain approximate solutions. As a person with a lot of friends may be regarded as influential, a widely adopted heuristic to address the influence maximization problem is to select seeds based on their degree, called *degree centrality*. Nevertheless, members of large communities often have a larger degree than members of small communities. Consequently, degree centrality may easily result in seeds in the same large community. As the influence spreads (i.e., the number of influenced nodes) of seeds in the same community tend to overlap, another commonly used heuristic is *distance centrality* which selects seeds in order of increasing average distance to other nodes. However, nodes in large communities usually have a small average distance, so distance centrality also results in seeds in the same large community. In summary, both degree centrality and distance centrality result in the phenomenon of seed clustering, leading to a sharp deterioration in influence spread.

The set cover greedy algorithm [Estevez et al. 2007], developed based on ICM, takes a priority to select the node with the highest "uncover degrees" as the seed. Once a seed 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 efficient under simple models, such as ICM. However, it has good influence spread only when the model has high success probability. The climbing-up greedy algorithm [Kempe et al. 2003] was proposed based on both 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. The potential-based node selection method [Wang and Feng 2009] is

proposed to select some inactive nodes that might not be optimal in the starting phase but which 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 the method in Kempe et al. [2003]. However, in practice, it is still not efficient enough for large-scale online social networks. Based on a variation of ICM, Saito et al. [2011] construct a layered graph approach and apply 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 for searching for multiple influential nodes. Finally, an efficient algorithm, based on the *degree discount heuristic*, was presented in Chen et al. [2009], which obtains approximate solutions in large datasets in only a few seconds. However, they work only under LTM or ICM. In addition, the degree discount heuristic is only for very low successful probability, that is, it is extremely hard for people to be influenced.

Borrowing the idea from Kempe et al. [2003], the enhanced greedy algorithm [Ma et al. 2008] is proposed for HDM. Like the climbing-up greedy algorithm, it incurs excessive computation and hence is not feasible for large-scale social networks.

Community Detection. A community is characterized as a subset of individuals who interact with each other more frequently than other individuals outside the community [Wasserman and Faust 1994]. Community discovery is similar but not equivalent to the conventional graph partitioning problem. Both community discovery and the graph partitioning problem aim to cluster vertices into groups. A key challenge for the former, however, is to decide what the "most natural" partition of a network is, that is, we do not need to give any heuristic information to guide the partition. Moreover, if there exists no good community structure, there is no need to partition the network. This is why we use the community detection algorithm rather than the graph partitioning algorithm in our research.

A quantitative measure, called modularity (Q), has been proposed [Wan et al. 2008] to assess the quality of community structures and to formulate community discovery as an optimization problem. Since optimizing Q is an NP-problem, several heuristic methods have been proposed, as surveyed in Danon et al. [2005]. Assume that M is the number of edges and N is the number of nodes. The time complexity of most community detection algorithms [Bortner and Han 2010; Danon et al. 2005; Ester et al. 1996; Feng et al. 2007; Girvan and Newman 2002; Huang et al. 2010; Lancichinetti et al. 2009; Ng et al. 2001; Newman 2004; Palla et al. 2005; Ruan and Zhang 2007; Wan et al. 2008; White and Smyth 2005; Xu et al. 2007] is between $O(N \log N)$ and $O(N^3)$.

3. THE COMMUNITY-BASED INFLUENCE MAXIMIZATION FRAMEWORK

In this section, we present the proposed community-based influence maximization (CIM) framework and discuss our approaches to address the arising issues. As illustrated earlier in Figure 2, CIM consists of three phases: (i) community detection; (ii) candidate generation; and (iii) seed selection. The following section details each phase of CIM.

3.1. Community Detection in CIM

In a social network, a community is a subset of individuals who interact with each other more closely than other individuals outside the community [Wasserman and Faust 1994]. As we observe some potential advantages of exploring community structures in influence maximization problem, designing an effective clustering algorithm for CIM is an immediate task we face. Such an algorithm needs to meet our ultimate goal of reducing computational overhead in the influence maximization problem is an immediate issue that we face. We believe that the notion and principles of community

are able to capture human nature in social networks. Thus, our community detection algorithm intends to detect the most natural communities of a social network without relying on heuristics, for example, the number of partitions. Notice that if a social network naturally accommodates three communities, we really should not force a partition of the network into four communities, even if an influence maximization task would like to select four seeds. Thus, in this article, to discover community structures, we aim to develop a clustering method without specifying the number of communities.

The clustering algorithm developed for phase (i) of CIM is called *hierarchical clustering* (abbreviated as *H_Clustering*). In H_Clustering, we incorporate the notion of *modularity* [Newman 2006] based on a bottom-up approach to iteratively merge nodes with strong structure similarity into communities. Initially, for each node in the given social network, H_Clustering derives the structural similarity between the node and its neighboring nodes, where the structure similarity is used as the edge weight for its neighboring nodes. The similarity function is defined as follows.

Definition 1 (*Structural Similarity between Nodes*). Given a social network $G = (V, E)$, the set of adjacent nodes of a node $u \in V$ is defined as $adj(u)$. Note that $adj(u)$ also includes u, that is, $u \in adj(u)$. The similarity between two adjacent nodes u *and* v is defined as follows:

$$Sim(u, v) = \frac{|adj(u) \cap adj(v)|}{\sqrt{|adj(u)| \times |adj(v)|}}. \tag{2}$$

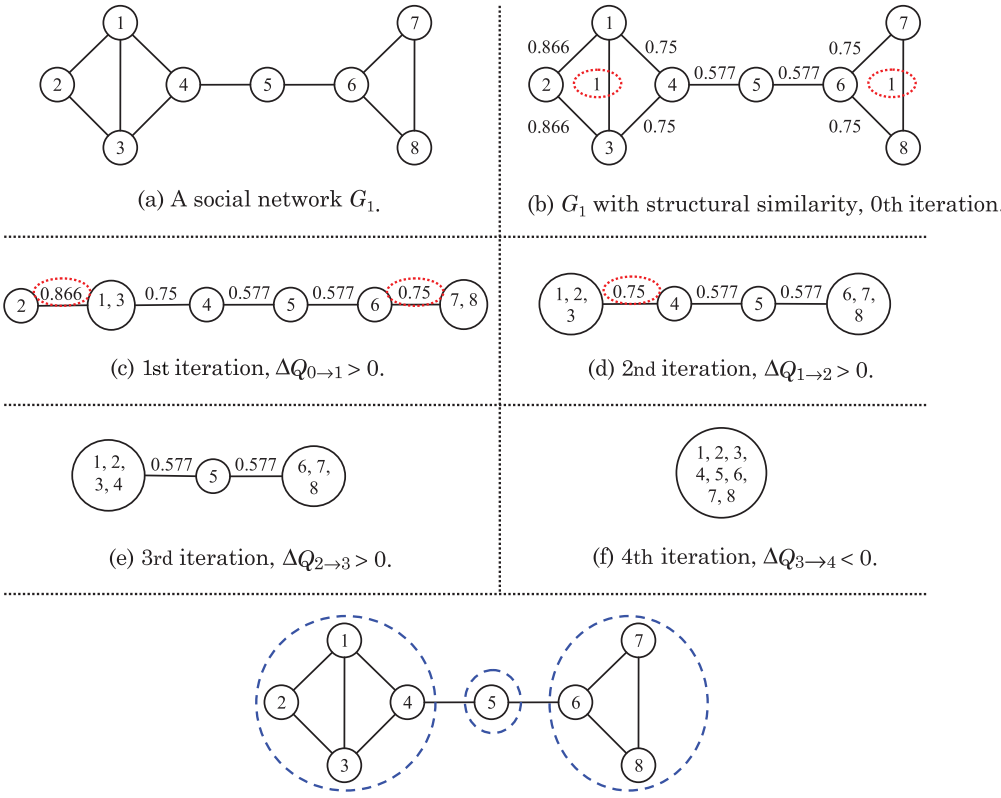
After obtaining the structure similarities of all edges in the network, H_Clustering first treats each node as a community and groups each pair of nodes into a community if the structural similarity between these two nodes is the largest among their surrounding edges from each other. For example, given two nodes u and v, if the edge (u, v) is the largest among all edges connecting to u and also is the largest among all edges connecting to v, we merge u and v into a community. Next, we treat each newly created community as a node, and the process continues until a termination condition is reached. Borrowing an idea from the SHRINK algorithm [Huang et al. 2010], we adopt the *modularity gain* [Feng et al. 2007; Wan et al. 2008] to measure the quality of discovered communities in order to decide when to stop the community detection process. The definition of modularity gain is as follows.

Definition 2 (*Modularity Gain*). Given a social network $G = (V, E)$ and its clustering result $C = \{c_1, c_2, \ldots, c_p\}$, the modularity function is defined as

$$Q(C) = \sum_{i=1}^{p} \left[\frac{IS_i}{TS} - \left(\frac{DS_i}{TS} \right)^2 \right], \tag{3}$$

where $IS_i = \Sigma_{u,v \in ci}\ Sim(u, v)$ is the summation of total similarity of nodes in cluster c_i, $DS_i = \Sigma_{u \in ci, v \in V}\ Sim(u, v)$ is the summation of similarity of nodes in cluster c_i and other nodes in the network, and $TS = \Sigma_{u,v \in V}\ Sim(u, v)$ is the summation of similarity between any two nodes in the network. For G, given two different clustering results C and C', the modularity gain from C to C' is defined as $\Delta Q_{C \to C'} = Q(C') - Q(C)$.

Notice that H_Clustering utilizes the modularity gain as the terminated criteria. At each iteration, based on the clustering result from the last iteration, we merge all pairs of nodes with the strongest structural similarity among their neighbors to form larger communities. Suppose the clustering result in the last iteration and in the current iteration are C and C', respectively. If the modularity gain from C to C' is negative, H_Clustering will stop clustering, since the previous clustering result is good enough. As a result, there are some *homeless* nodes in the network which are not included in any community. Similar with the SHRINK algorithm, these homeless nodes represent two



(a) A social network G_1.

(b) G_1 with structural similarity, 0th iteration.

(c) 1st iteration, $\Delta Q_{0\to1} > 0$.

(d) 2nd iteration, $\Delta Q_{1\to2} > 0$.

(e) 3rd iteration, $\Delta Q_{2\to3} > 0$.

(f) 4th iteration, $\Delta Q_{3\to4} < 0$.

(g) G_1 with community structure.

Fig. 4. An example to show community discovery by H_Clustering.

interesting node types: hub and outlier. The homeless nodes whose neighbors are within only one community are outliers; other homeless nodes are hubs that connect different communities. Hubs can provide more information about the community structures and the connectivity of networks, which are very useful for influence maximization. We will discuss this in the next section. To justify the proposed H_Clustering, we compare H_Clustering with several clustering algorithms [Huang et al. 2010; Karypis and Kumar 1996; Newman 2002, 2004; Ruan and Zhang 2007] in Section 4.3 to show the runtime performance and how the discovered communities affect the influence spread in CIM.

Figure 4 illustrates an example of discovering communities by H_Clustering. Given a network G_1, H_Clustering first derives the structural similarity of all edges in G_1 with Eq. (2), as shown in Figure 4(b). We use edge (4, 5) as an example. Since $adj(4) = \{1, 3, 4, 5\}$ and $adj(5) = \{4, 5, 6\}$, $Sim(4, 5) = \frac{|\{4,5\}|}{\sqrt{|\{1,3,4,5\}|\times|\{4,5,6\}|}} = \frac{2}{\sqrt{4\times3}} = 0.577$. Other edge weights in the example are similarly obtained. At each iteration, we merge each pair of nodes into a community if both of these two nodes have the largest similarity among their other neighbors from each other, that is, the similarity of the edge between two nodes is not less than their surrounding edges. As the two edges in red dotted circles, as shown in Figure 4(b), are the largest, we can group them into communities, resulting in a new graph, as shown in Figure 4(c). Repeating the process, we obtain

Figure 4(d) by merging node 2 with node $\{1, 3\}$ and node 6 with node $\{7, 8\}$, respectively. Similarly, node $\{1, 2, 3\}$ and node 4 are merged to obtain the graph in Figure 4(e). H_Clustering stops at the 4th iteration after making an attempt to merge all nodes into one community in Figure 4(f), because $\Delta Q_{3\to4} < 0$. Consequently, H_Clustering outputs the discovered community structure, as shown in Figure 4(g), where node 5 is a hub connecting communities $\{1, 2, 3, 4\}$ and $\{6, 7, 8\}$.

3.2. Candidate Generation

In light of the discovered community structures, the candidate generation phase aims to determine a set of candidate seeds based on the size of communities and the connectivity of the nodes among communities. Notice that as social networks in realistic settings are extremely large, the search space for selecting seeds with maximal influence spread is also huge. Therefore, there is a need to effectively reduce the number of candidate seeds. How to narrow down the size of the candidate set of seeds is a core issue in CIM, faced in this phase.

Based on our observations discussed in Section 2.2, seeds selected from a large community could trigger more adoptions of a product or an innovation than seeds selected from a small community. Therefore, an intuitive approach is to select the centroid nodes of the k-largest communities in the social network as the k-influential seeds. However, this naïve approach has some potential problems: (1) the aforementioned observation may also imply that we should select more seeds in large communities; (2) some valuable information in the community structure is ignored. Notice that the proposed H_Clustering, in addition to detecting communities, also identifies hubs (i.e., a node connecting different communities). While the centroids of communities may seem to be natural candidates for seed selection, the hubs should also be considered, as they can easily spread their influences to many different communities.

Instead of simply designating an arbitrary number of large communities as significant, we formally define *significant communities* as those that have the number of nodes larger than the average number of nodes a seed may influence in a given influence maximization task. By pruning the insignificant communities, we can effectively reduce the number of seed candidates.

Definition 3 (*Significant Communities*). Given a social network $G = (V, E)$ and a number k, H_Clustering derives a set of communities, denoted as $C = \{c_1, c_2, \ldots, c_p\}$, where c_i is the ith community and n_i denotes the number of nodes in c_i. The set of significant community C_s is defined as follows:

$$C_s = \left\{ c_j \in C | n_j \geq \frac{\sum_{i=1}^{p} n_i}{k} \right\}. \tag{4}$$

How to decide the centroids of significant communities is also an essential problem. In general, the degree of nodes in a social network fits the power-law distribution [Albert et al. 1999; Barabasi and Albert 1999], that is, a very large number of nodes have a very small number of neighbors. High-degree nodes usually reside in large communities. Hence, our first strategy is to consider high-degree nodes as the centroids of communities. However, some information obtained in phase (i) of CIM is also useful for identifying the centroids of community. Since the similarity score defined in Definition 1 reveals the structural information of a node, the information is also a good criterion to leverage for selecting the centroids of communities. The summation of these scores of a node is a potential factor for candidate generation. Our second strategy is to identify the nodes with large summation of similarity scores as the centroids of communities.

With the purpose of generating a small set of seed candidates (instead of exhaustively considering all the nodes in the network), our method is to consider only the potential centroid nodes (i.e., nodes with high degree and large score sum) in significant communities and the hub nodes as candidates by eliminating outliers and nodes in some "insignificant communities". Accordingly, we collect the top-p% of high-degree nodes and large-score-sum nodes in each significant community and all hub nodes connecting significant communities as the candidate set. Our empirical study (detailed in Section 4.4) shows that, in most cases, including the top-10% of nodes of each significant community, the candidate set is sufficient to select good seed nodes. For example, as in Figure 3, community 3 is an insignificant community and will be pruned off. Suppose we collect the top-20% of high-degree nodes in each significant community. The candidate set consists of three nodes: nodes 3 and 6 in community 1, node 13 in community 2, and hub node 10 connecting communities 1 and 2.

3.3. Seed Selection

Given a set of candidate seeds, phase (iii) of CIM is to decide k of them as the final seeds. Although we have effectively reduced the number of seed candidates, the task of seed selection in algorithm CIM is still very time consuming (albeit it has already improved a lot). Specifically, for various combinations of k seeds selected from the candidate seed set, the computation of influence spreads is carried out by simulating how influences (heat) spread from those seeds based on the heat diffusion model that incurs a significant amount of computation cost. We address this problem with a two-step approach for seed selection. In the first step, we adopt a quota-based approach to determine the number of seeds to be allocated for a given significant community and then determine which nodes should be selected for this significant community based on a heuristic of position score and hub purity. As this is a fast heuristic-based seed selection approach, we do not expect the result to be optimal. Nevertheless, we believe that the selection has its merits and can be used to filter out the majority of inferior selections. Thus, in the second step, we heuristically, again based on the community structures, find a new candidate seed which may potentially increase the influence spread to swap with a seed node, aiming to obtain a better seed set. This process is repeated until the influence spread does not improve any more or a certain number of seed node swapping has been performed.

Here we first describe how we obtain the initial seed selection in step 1. As the size of communities is a factor for determining significant communities, we shall not treat all the significant communities the same, as placing seeds in a large community could trigger more adoptions of a product or an innovation than in a small community. CIM uses the total number of nodes in significant communities to decide how many seeds are allocated to each significant community. Given a social network $G = (V, E)$ and a number k, let C_s be a set of significant communities in G. Suppose $C_s = \{c'_1, c'_2, \ldots, c'_q\}$ and each c'_i has n'_i number of nodes; the quota of seed nodes allocated in c'_i is defined as

$$quota(c'_i) = k \times \frac{n'_i}{\sum_{j=1}^{q} n'_j}. \tag{5}$$

Consider the example in Figure 3, where the number of seeds is set to three. From our formulas in Eq. (5), communities 1 and 2 will be allocated two seeds and one seed, respectively.

After deciding the quota for communities in C_s, we take turns to select $quota(c'_i)$ of *high-priority* nodes for each community c'_i. Notice that if we only consider the member nodes of a significant community for its allocated quota, we shall leave out those important hub nodes. Thus, we include all the candidate nodes of c'_i (i.e., the top-p%

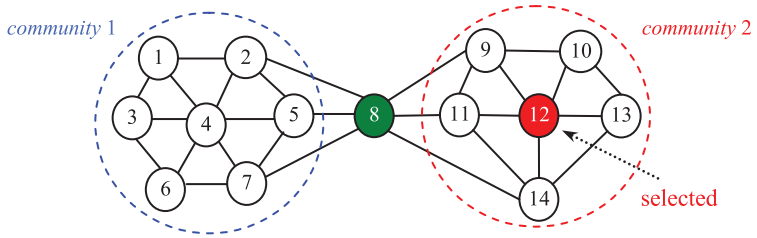


Fig. 5. An illustration of position score and hub purity.

nodes with high degree and large score sum) and all the hub nodes connecting to c_i', to compete for the quota of c_i'. The selected seeds for c_i' are included in the *initial* seed set (for further tuning in the second step).

In the following, we detail some concepts/notions used in the seed selection of CIM. Note that the main goal in phase (iii) is to set the priority among the high-degree nodes, large-score-sum nodes, and hub nodes to select seeds. Hence, we first define two metrics, *position_score* (in Definition 4) and *hub_purity* (in Definition 5), for hub nodes, because their position in the network bears importance. Also, hubs connecting to too many communities may possibly impair their influence. Then, we formulate a *compare_priority* function (in Definition 6) to decide the priority of two nodes in the candidate set. Finally, two evaluation metrics, *left* and *seed load*, are defined (in Definitions 7 and 8), serving as the heuristics to determine the final seeds, respectively.

Definition 4 (*Position Score*). Given a social network $G = (V, E)$ and a set of significant communities $C = \{c_1, c_2, \ldots, c_p\}$, to assess the importance of a hub's position in the network, the position score of a hub $u \in V$ is defined as the number of communities which u connects to, that is,

$$position\_score(u) = |\{c_i | \exists (u, v) \in E \wedge v \in c_i\}|. \tag{6}$$

Obviously, if node u is a hub node, the $position\_score(u)$ is greater than 1; otherwise, the $position\_score(u)$ is 1. In general, the more communities connected to a hub, the more easily influence propagates from the hub to these communities. However, if some connected communities of a hub have chosen their seeds, the influential power of the hub may deteriorate. Thus, *hub_purity* of a hub is defined to provide a precise measure of the remaining importance of a hub node.

Definition 5 (*Hub Purity*). Given a social network $G = (V, E)$ and a set of significant communities $C = \{c_1, c_2, \ldots, c_p\}$, for a hub $u \in V$,

$$hub\_Purity(u) = \frac{|\{c_i | \exists (u, v) \in E \wedge v \in c_i \wedge c_i \notin SC|}{position\_score(u)}, \tag{7}$$

where SC is a set of communities in C that have already chosen initial seeds. For example, in Figure 5, node 8 is a hub connecting community 1 and community 2; its *position_score* is 2 (because it connects two communities). If community 2 has selected node 12 as an initial seed, its *hub_purity* is 1/2.

Notice that a high-priority node refers to a node that has a high degree in its community, possesses a large-summation similarity score in its community, or is in a good position in the network. To decide the priority of two nodes in a candidate set, we use the function *compare_priority*, which works as follows.

Definition 6 (*Compare_priority*). Given two candidate nodes of a community, which can be a high-degree node in the community, large-score-sum node in the community,

or a hub node connecting to the community, the priority order of these two nodes is governed by the following rules.

(1) To compare two non-hub nodes, the node with a higher degree has a higher priority. If two non-hub nodes have the same degree, we choose the node with a larger summation of similarity score. Because the position of a non-hub node is not as critical as a hub node, we compare two non-hub nodes by their influences on their neighbors (i.e., the degree and the sum of similarity scores).
(2) To compare two hub nodes, the node with a higher position_score has a higher priority. If two hub nodes have the same position_score, we choose the one with a higher degree; and if two hub nodes have the same position_score and degree, we choose the one with a larger summation of similarity score. We compare hubs according to how many communities they belong to, since we want to choose a hub which is in an important position in the network. Besides, the purities of comparing hubs must exceed the purity threshold. A low-purity hub is not a good choice, since it may cover too many communities.
(3) To compare a non-hub node with a hub node, the node with a higher degree has a higher priority; and if two nodes have the same degree, the node with a larger score summation has a higher priority. If the two nodes have the same degree and score summation, the hub node has a higher priority, since it can spread influence among many communities.

Recall our example in Figure 3, where three seeds are to be selected; as mentioned in Section 3.2, the candidate set of community 1 includes nodes 3, 6, and 10, and the quota of community 1 is two. With the compare_priority function, we will select nodes 3 and 6 as the initial seeds of community 1.

By collecting all selected top-$quota(c_i')$ *high-priority* nodes in each significant community c_i', we can derive a set of initial seeds. The initial seeds may be the final seeds; however, not all of them can derive good influence spread under different states of the heat diffusion model, that is, some seeds may perform well within a short period after the heat (influence) is injected, but they may not perform well for a long run. In other words, the parameter setting in the heat diffusion model has an impact on selecting final seeds. Hence, in the second step of phase (iii), CIM heuristically replaces initial seeds with the nodes remaining in the candidate set to test whether we can increase the influence spread. This process allows us to tune the influence spread under different parameter settings in HDM, including flow duration, activation threshold, and thermal conductivity. We discuss the effect of flow duration, activation threshold, and thermal conductivity individually.

Comparing the different effect 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 better 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 one community with a high activation threshold. High thermal conductivity makes information diffuse more quickly. Compared with low thermal conductivity, information of high thermal conductivity makes information diffuse over a longer distance. Hence, we do not select many seeds from one community with high thermal conductivity. We conclude that different parameter settings may have an impact on seed selection. It is better to select more seeds in one community with a short flow duration, high threshold, and low thermal conductivity. On the contrary, in social networks with long flow duration, low threshold, and high thermal conductivity, not many

seeds in a single community are needed. Therefore, the tuning in CIM by swapping seeds allows us to test and verify whether large communities should need more seeds.

Here we describe the tuning process. CIM swaps some selected initial seeds to identify the final seeds. The tuning takes r iterations to test the swapping heuristically. Usually, $r = k \sim 2k$ is sufficient to obtain satisfactory influence spread. We choose a new seed (called *add-node*) to replace an existing seed node (called *delete-node*) to test whether the influence spread increases after the swapping. Two evaluation metrics, *left* and *seed load*, are defined as the heuristics to determine the add-nodes and delete-nodes.

Definition 7 (*Left*). Given a social network $G = (V, E)$ and a set of significant communities $C_s = \{c'_1, c'_2, \ldots, c'_q\}$, suppose that we have selected a set of initial seeds, PS. After running HDM on G with PS, we obtain a set of active nodes $I_{PS}(G)$. We define a function,

$$left(c_i{}') = |\{u | u \in c_i{}' \text{ and } u \text{ is a non-activated node}\}|. \tag{8}$$

left (c'_i) may be seen as the potential nodes to be influenced and thus the potential gain by adding more seeds to c'_i. As *left* (c'_i) increases, the potential gain by selecting more seeds in c'_i also increases.

On the other hand, we define a function,

Definition 8 (*Seed Load*). Given a social network $G = (V, E)$ and a set of significant communities $C_s = \{c'_1, c'_2, \ldots, c'_q\}$,

$$seed\_load(c'_i) = \frac{\text{size}(c'_i)}{|\{u | u \in c'_i \cap PS\}|}. \tag{9}$$

The function $seed\_load(c'_i)$ indicates whether too many seeds are selected from c'_i. When $seed\_load(c'_i)$ is small, there are too many seeds in c'_i.

In each iteration ℓ (where $1 \leq \ell \leq r$), CIM first selects the node with the maximal priority node from community c'_i in C_s which has ℓ-largest $left(c'_i)$ as an *add-node*. Note that the add-node is certainly not a seed node in PS. Since we have already collected all of the top-p% highest-degree nodes and largest-score-sum nodes of c'_i and all hub nodes connecting c'_i to form the candidate set, add-node can be found quickly. Then CIM selects a delete node, *delete-node*, from PS, where the delete-node is the minimum priority node in c'_j which has minimum $seed\_load(c'_j)$. Finally, we test whether we should substitute an add-node for the delete-node. If the influence spread after swapping increases, we formally make the swap and continue the process. In general, $r = k \sim 2k$ is sufficient to obtain satisfactory influence spread in most cases.

3.4. CIM Algorithm

Based on the preceding descriptions for each phase, the algorithm for the whole CIM framework is shown in Algorithm 1. The information of community structure and hubs can be discovered by H_Clustering. With insignificant community pruning, CIM can derive a set of significant communities (lines 2–3). Suppose we have a set of significant communities $C_s = \{c'_1, c'_2, \ldots, c'_q\}$, as defined in Definition 3. For each c'_i, we select the top-p% high-degree nodes and large-score-sum nodes in c'_i and hub nodes connecting to c'_i to form a $candi\_set.c'_i$ (lines 4–5). Then we choose top-$quota(c'_i)$ high-priority nodes from the candidate set as initial seed nodes (lines 6–13). Note that if we have selected a hub node connecting to c'_i as an initial seed, each unchosen significant community which has an edge to this hub requires reducing its size (lines 11–13). For example, as shown in Figure 5, if node 8 had been selected as an initial seed node in community 1 (size = 7), the size of community 2 (size = 6) would reduce the degree of node 8 in community 2, that is, $6 - 3 = 3$. Community-size reduction can effectively avoid

ALGORITHM 1: *CIM* (G, k)

Input: Graph of social network *G(V, E)*; number of total seeds k
Output: k seeds

01: $PS \leftarrow \emptyset$; // set of initial seeds

// phase (i): community detection
02: $C = \{c_1, c_2, \ldots, c_p\} \leftarrow H\_Clustering(G)$;
03: $C_s = \{c'_1, c'_2, \ldots, c'_q\} \leftarrow$ pruning insignificant community in C;

// phase (ii): candidate generation
04: **for each** c'_i in C_s **do**
05: $candi\_set.c'_i \leftarrow$ collect top-p degree nodes and score-sum nodes of c'_i and hub nodes connecting to ci';

// phase (iii): seed selection
06: **for each** c'_i in C_s **do** // select the initial seed nodes
07: $n \leftarrow quato(c'_i)$;
08: $prio\_set \leftarrow compare\_priority(candi\_set.c'_i)$ and select top-n priority nodes;
09: $PS \leftarrow PS \cap prio\_set$;
10: **If** $\exists$ hub node h in $prio\_set$ **then** // size reduction
11: **for each** c'_j in C_s which has edge connected to h **do**
12: $m \leftarrow |\{u | u \in c'_j \wedge \exists(u, h) \in E\}|$; // number of nodes in c'_j connected to h
13: $size(c'_j) \leftarrow size(c'_j) - m$;
14: $I_{PS}(G) \leftarrow$ execute HDM on G with PS; // $I_{PS}(G)$: the set of active nodes of PS
15: $IM \leftarrow |I_{PS}(G)|$; // IM : save the max number of active nodes

16: **for** $\ell = 1$ to r **do** // initial seed tuning
17: $T \leftarrow PS$; // *a temp set*
18: $add\_node \leftarrow$ select maximum priority node $u \in candi\_set.c'_i$ where *left* (c_i) is top-ℓ maximum in C_s;
19: $delete\_node \leftarrow$ select minimum priority node $v \in c'_j \cap PS$ where $seed\_load(c'_j)$ is minimum in Cs;
20: replace the $delete\_node$ with add_node in T;
21: $I_T(G) \leftarrow$ execute HDM on G with T;
22: **if** $I_T(G) > IM$ **then**
23 $PS \leftarrow T; IM \leftarrow |I_T(G)|$;
24: Output nodes in PS as seed nodes;

influence overlap, since size reduction also changes the quota of allocated seed nodes in the community. The initial seed nodes may have a very good position connecting many significant communities or have high connectivity in a community. The set of selected initial seeds is denoted as PS. Finally, CIM tunes selected initial seeds to identify the final seeds (lines 16–23) and outputs the selected nodes as seeds (line 24).

4. EXPERIMENTS

To evaluate the performance of proposed CIM, three influence maximization algorithms, CDH-Kcut [Chen et al. 2012], *CDH-SHRINK* [Chen et al. 2012], and enhanced greedy algorithm (EGA) [Ma et al. 2008], 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 heat diffusion model, are implemented in the C++ language, and are tested on a Pentium D 3.0GHz with 2GB of main memory running the Windows XP system. The

Table I. Parameters of LFR Benchmark Graphs

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 II. 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

comprehensive performance study is conducted on five synthetic networks and three real-world datasets, the karate network [Zachary 1997], the NETHep network [Chen et al. 2012], and the Facebook network [Chen et al. 2012]. In each experiment, we vary some parameters of the heat diffusion model to compare the influence spread (number of activated nodes) and efficiency (execution time) of the five algorithms.

4.1. Synthetic Networks

The synthetic datasets in the experiments are generated using a synthetic generation program, Lancichinetti-Fortunato-Radicchi (LFR) benchmark graphs [Lancichinetti et al. 2008]. The parameter settings of the LFR generator are shown in Table I. We generated five different undirected graphs, (1) 1000Smp: the graph with 1,000 nodes and small mixing parameter; (2) 1000Lmp: the graph with 1,000 nodes and large mixing parameter; (3) 1000$Lmaxd$: the graph with 1,000 nodes and large maximum degree; (4) 1000LM: the graph with 1,000 nodes and large number of degree; and (5) 5000Smp: the graph with 5,000 nodes and small mixing parameter, as shown in Table II. Generally speaking, the higher the mixing parameter of a network, the more difficult it is to find the community structure. Since all generated graphs are very small, the experiments only discuss the influence spread. The comparison of execution time will be performed in the next section (on NETHep network and the Facebook network).

Table III shows the results of the five algorithms with different activation thresholds (θ), flow duration (t), and thermal conductivity (α) on 1000Smp. We can observe that CIM almost has the best influence spread. In the case $t = 0.3$, $\theta = 0.1$, $\alpha = 0.1$, the number of active nodes is even larger than EGA. We find an interesting result; a higher activation threshold leads to the phenomenon of seed clustering. Hence, by the community comparison and the allocation of seed quota, CIM can derive a better influence spread. Table IV shows the influence spread of different algorithms with different parameter settings on 1000Lmp. If it is difficult to identify the correct community structure of the network, CIM performs worse than EGA; however, the number of active nodes is better than CDH-SHRINK and CDH-Kcut. The accuracy of the detected community structure reflects the performance of the influence spread of CIM. Therefore, with increased mixing parameters (i.e., 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 V indicates the influence spread of different algorithms with different parameter settings on 1000$Lmaxd$. Nodes in 1000$Lmaxd$ have larger degrees, that is, the

Table III. The Influence Spread of Different Algorithms on 1000Smp

parameter setting	DH	EGA	CDH (SHRINK)	CDH (Kcut)	CIM
$t = 0.1, \theta = 0.1, \alpha = 0.1$	215	**341**	**341**	339	**341**
$t = 0.1, \theta = 0.2, \alpha = 0.1$	215	336	335	332	**343**
$t = 0.2, \theta = 0.1, \alpha = 0.1$	336	**386**	379	345	383
$t = 0.3, \theta = 0.1, \alpha = 0.1$	472	503	499	**508**	**508**
$t = 0.1, \theta = 0.1, \alpha = 0.2$	336	**386**	**386**	345	**386**
$t = 0.1, \theta = 0.1, \alpha = 0.3$	472	**503**	498	**503**	**503**

Note: The largest influence spread is highlighted in boldface.

Table IV. The Influence Spread of Different Algorithms on 1000Lmp

parameter setting	DH	EGA	CDH (SHRINK)	CDH (Kcut)	CIM
$t = 0.1, \theta = 0.1, \alpha = 0.1$	206	**251**	249	233	**251**
$t = 0.1, \theta = 0.2, \alpha = 0.1$	187	**225**	**225**	215	**225**
$t = 0.2, \theta = 0.1, \alpha = 0.1$	484	**562**	559	535	560
$t = 0.3, \theta = 0.1, \alpha = 0.1$	722	**790**	782	763	787
$t = 0.1, \theta = 0.1, \alpha = 0.2$	484	**562**	560	520	**562**
$t = 0.1, \theta = 0.1, \alpha = 0.3$	722	**790**	781	742	786

Note: The largest influence spread is highlighted in boldface.

Table V. The Influence Spread of Different Algorithms on 1000$Lmaxd$

parameter setting	DH	EGA	CDH (SHRINK)	CDH (Kcut)	CIM
$t = 0.1, \theta = 0.1, \alpha = 0.1$	202	332	333	315	**335**
$t = 0.1, \theta = 0.2, \alpha = 0.1$	196	290	288	269	**292**
$t = 0.2, \theta = 0.1, \alpha = 0.1$	299	494	**495**	473	**495**
$t = 0.3, \theta = 0.1, \alpha = 0.1$	404	565	569	561	**573**
$t = 0.1, \theta = 0.1, \alpha = 0.2$	299	494	**495**	473	**495**
$t = 0.1, \theta = 0.1, \alpha = 0.3$	404	565	569	561	**573**

Note: The largest influence spread is highlighted in boldface.

degree of some nodes will be much larger than that of others. The community structure in this graph is clearly manifested. Consequently, the performance of the influence spread of two CDHs and CIM will improve, especially with a high activation threshold. Note that the result of DH is also improved. By the observation, we can see that CIM has the best influence spread with all parameter settings, because it can find a very good community structure and allocate the correct number of seeds in each significant community. Although CDHs (based on SHRINK or Kcut) also discover the community structure, they omit the size effect and just give every community the same number of seeds.

Table VI shows the influence spread of five algorithms with different parameters on 1000LM. In 1000LM, each node has more neighbors, so information will spread quickly. Hence, we could see that the influence spread in 1000LM is higher than that in 1000Smp, 1000Lmp, and 1000$Lmaxd$. In most cases, the influence spreads of CIM are the same as EGA and better than CDH-SHRINK, CDH-Kcut, and DH. Table VII indicates the influence spread of different algorithms with varying activation thresholds, flow duration, and thermal conductivity on 5000Smp. 5000Smp is larger than these graphs. Obviously, in most cases, the influence spread of CIM is almost the same as EGA and still better than CDH-SHRINK, CDH-Kcut, and DH. In summary, CIM has almost as good an influence spread as EGA. Considering the size effect and insignificant pruning of the community, CIM is more productive than both CDH-SHRINK and CDH-Kcut.

Table VI. The Influence Spread of Different Algorithms in 1000LM

parameter setting	DH	EGA	CDH (SHRINK)	CDH (Kcut)	CIM
$t = 0.1, \theta = 0.1, \alpha = 0.1$	521	**663**	653	653	**660**
$t = 0.1, \theta = 0.2, \alpha = 0.1$	316	**506**	499	487	**506**
$t = 0.2, \theta = 0.1, \alpha = 0.1$	184	**226**	**226**	218	**226**
$t = 0.3, \theta = 0.1, \alpha = 0.1$	816	**923**	919	898	920
$t = 0.1, \theta = 0.1, \alpha = 0.2$	996	**1,000**	993	**1,000**	**1,000**
$t = 0.1, \theta = 0.1, \alpha = 0.3$	816	**923**	919	898	920

Note: The largest influence spread is highlighted in boldface.

Table VII. The Influence Spread of Different Algorithms in 5000Smp

parameter setting	DH	EGA	CDH (SHRINK)	CDH (Kcut)	CIM
$t = 0.1, \theta = 0.1, \alpha = 0.1$	391	**540**	**540**	536	**540**
$t = 0.1, \theta = 0.2, \alpha = 0.1$	252	**438**	433	426	436
$t = 0.2, \theta = 0.1, \alpha = 0.1$	169	**261**	255	232	259
$t = 0.3, \theta = 0.1, \alpha = 0.1$	843	**1,258**	1,238	1,202	**1,258**
$t = 0.1, \theta = 0.1, \alpha = 0.2$	952	**1,451**	1,423	1,378	**1,451**
$t = 0.1, \theta = 0.1, \alpha = 0.3$	843	**1,258**	1,238	1,202	**1,258**

Note: The largest influence spread is highlighted in boldface.

4.2. Real Datasets

In addition to using synthetic datasets, we have also performed some experiments on three real datasets: (1) Zachary's karate network, (2) the NETHep network, (3) the and Facebook network, to compare the performance and indicate the applicability of CIM algorithm.

4.2.1. Zachary's Karate Network. Zachary's network [Zachary 1997] 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 three years. Edges connect individuals who have been observed to interact outside the activities of the club. The Zachary's karate network is shown in Figure 6. Table VIII lists the influence spread and the selected seeds (the two numbers in parentheses) of the five algorithms with different parameter settings. From Table VIII, we can find that CIM could select good seeds according to different parameter values. In every case, CIM can obtain the same influence spread as EGA. Furthermore, in the case $t = 0.4$, $\theta = 0.6$, $\alpha = 0.1$, the CIM obtains better influence spread than EGA. In the case $t = 0.1$, $\theta = 0.2$, $\alpha = 0.1$ and case $t = 0.1$, $\theta = 0.2$, $\alpha = 0.2$, two seeds, 0 and 33, are selected, as the red nodes in Figure 6. However, we could see that the two seeds selected in case $t = 0.1$, $\theta = 0.2$, $\alpha = 0.1$ are 32 and 33. This is the reason why a high activation threshold easily causes the phenomenon of seed clustering but high thermal conductivity does not. Therefore, two seeds 0 and 33 are selected in case $t = 0.1$, $\theta = 0.2$, $\alpha = 0.2$, which are the same selections in case $t = 0.1$, $\theta = 0.1$, $\alpha = 0.1$.

4.2.2. NETHep Network. In this section, we extract a large real academic collaboration network, NETHep (`http://snap.stanfords.edu/data`), from the e-print. Each node in the network represents an author. If an author a co-authored a paper with author b, the graph contains an undirected edge from a to b. If the paper is co-authored by n authors, this generates a completely connected graph of n 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 discuss the efficiency and influence spread of CIM, EGA, CDH-SHRINK, CDH-Kcut, and DH with different numbers

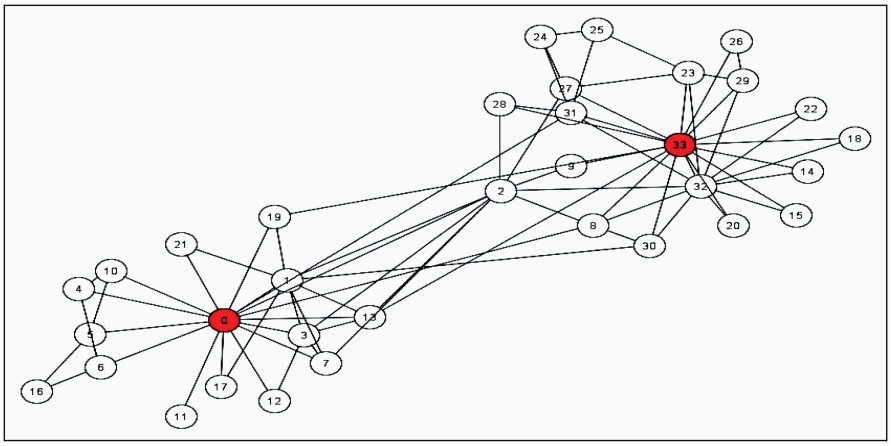


Fig. 6. Zachary's karate Network (red nodes are node 0 and node 33).

Table VIII. The Influence Spread and Two Selected Seeds of Different Algorithms in Zachary's Karate Network

parameter setting	DH	EGA	CDH (SHRINK)	CDH (Kcut)	CIM
$t = 0.1, \theta = 0.1, \alpha = 0.1$	**31 (0, 33)**	**31 (0, 33)**	**31 (0, 33)**	**31 (0, 33)**	**31 (0, 33)**
$t = 0.1, \theta = 0.2, \alpha = 0.1$	6 (0, 33)	**12 (32, 33)**	**12 (32, 33)**	**12 (32, 33)**	**12 (32, 33)**
$t = 0.1, \theta = 0.3, \alpha = 0.1$	6 (0, 33)	**12 (32, 33)**	**12 (32, 33)**	**12 (32, 33)**	**12 (32, 33)**
$t = 0.1, \theta = 0.2, \alpha = 0.2$	**31 (0, 33)**	**31 (0, 33)**	**31 (0, 33)**	**31 (0, 33)**	**31 (0, 33)**
$t = 0.4, \theta = 0.6, \alpha = 0.1$	6 (0, 33)	8 (4, 7)	**12 (32, 33)**	**12 (32, 33)**	**12 (32, 33)**

Note: The largest influence spread is highlighted in boldface.

of seeds and values of parameters. In CIM, the purity threshold is set to 0.3, which can get satisfactory influence spread. Figure 7 shows the influence spread of different algorithms with different numbers of seeds on NETHep. The x-axis indicates the number of seeds and the y-axis indicates influence spread. In most cases, CIM's influence spread $\approx$ EGA's influence spread $>$ CDH-SHRINK's influence spread $>$ CDH-Kcut's influence spread $>$ DH's influence spread. With the increasing number of seeds, the improvement of influence spread of CIM is better than that of two CDH algorithms. Note that DH has poor results, since most seeds selected by DH are only in a few communities.

Figures 8(a) and 8(b) show the influence spread of 10 seeds and 30 seeds with different activation threshold (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 show that although the total influence spread of the four algorithms will decrease as θ increases, CIM still maintains 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 CIM when selecting more seeds.

Figures 9(a) and 9(b) indicate the influence spread of 10 seeds and 30 seeds with different flow duration (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 algorithm still maintains 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 five algorithms.

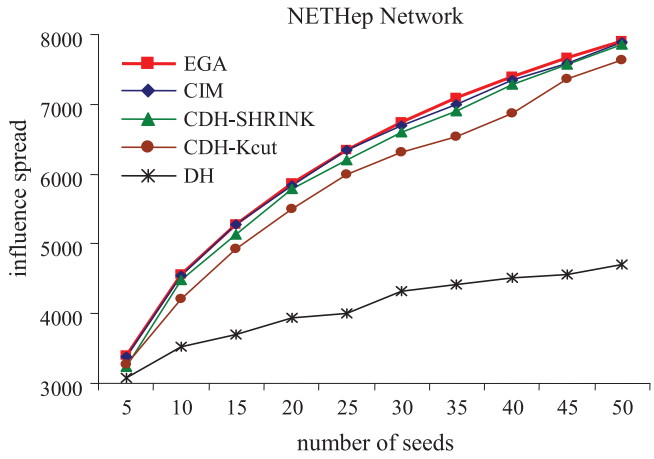
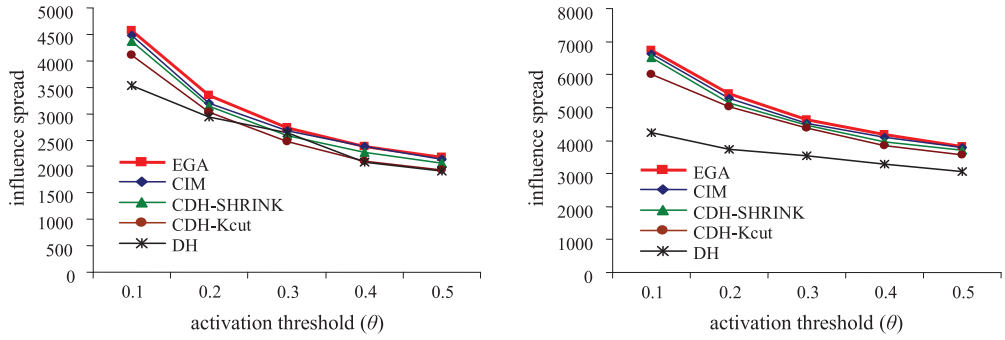


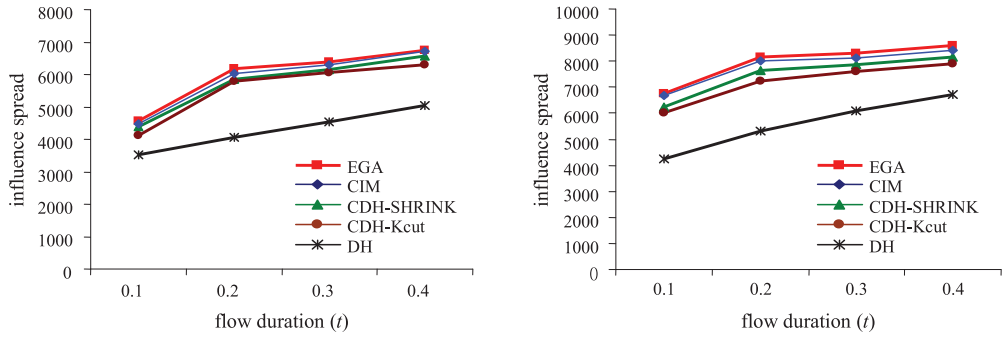
Fig. 7. Influence spread of five algorithms on NETHep with $t = 0.1, \theta = 0.1, \alpha = 0.1$.



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

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

Fig. 8. Influence spread of different algorithms on NETHep with different activation threshold.



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

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

Fig. 9. Influence spread of different algorithms on NETHep with different flow durations.

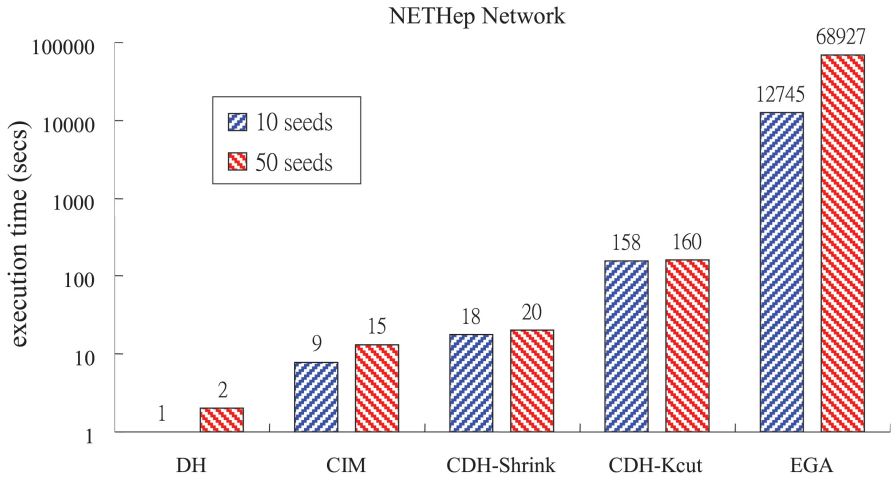


Fig. 10. Running time of different algorithms on the NETHep network when selecting 10 seeds and 50 seeds, respectively.

Figure 10 shows the execution time of five algorithms 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 small. CIM has the most efficient execution time among the other three algorithms and is about 4,924 times faster than EGA (68927/14). We can also observe that the running time of EGA is proportional to the number of seeds. When selecting 10 seeds and 50 seeds, the execution times of CIM are only slightly different. This is because CIM only has to spend a little more time on turning initial seeds when the number of nodes increases. As shown in Figures 7 and 10, although on average EGA is about 0.8% better than CIM in terms of influence spread, the execution time is much slower than the proposed CIM algorithm. According to the observation, compared with two CDH algorithms, CIM can effectively reduce the iteration of initial seed tuning, because it can find a very good community structure and allocate the correct number of seeds in each significant community. Although CDHs (based on SHRINK or Kcut) also discover community structure, they omit the size effect and just give every community the same number of seeds. CIM also reduces the search space of seed identification with the proposed insignificant pruning method. Nowadays, social networks are becoming bigger and bigger. If an 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.

4.2.3. Facebook Network. In this section, we discuss the influence maximization on a large real social dataset, the Facebook network [Chen et al. 2012]. Each node in the network represents a user. If user a is a friend of user b, the graph contains an undirected edge from a to b. The network is denoted as FB, in the period from April 2004 to January 2009 (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 CIM, the purity threshold is set to 0.2, which can demonstrate satisfactory influence spread in FB.

Figure 11 shows the influence spread of five 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 the figure, in most cases, CIM's influence spread $\approx$ EGA's

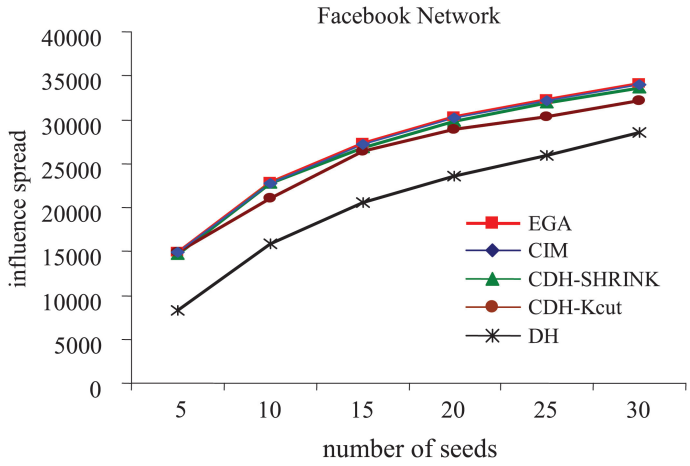
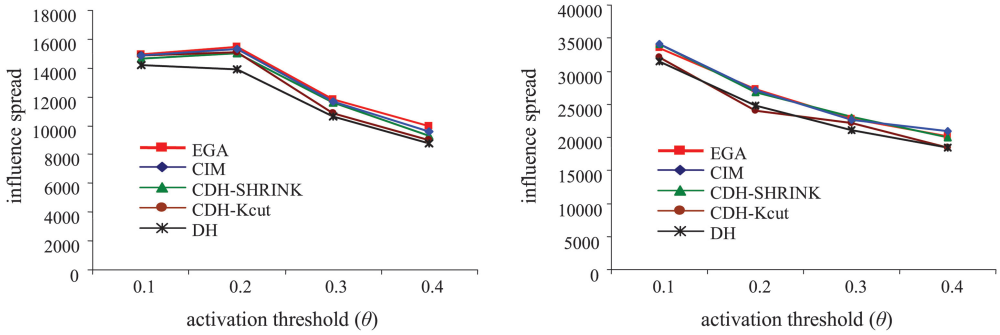


Fig. 11. Influence spread of five algorithms on Facebook with $t = 0.1, \theta = 0.1, \alpha = 0.1$.



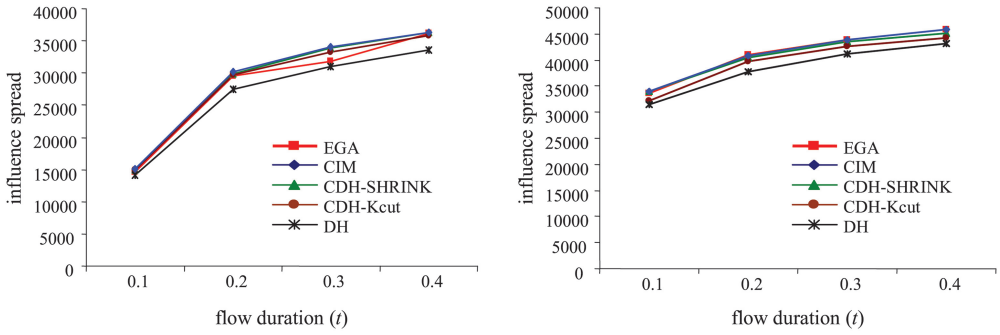
(a) Influence spread of 10 seeds with different activation threshold.

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

Fig. 12. Influence spread of different algorithms on Facebook with different activation threshold.

influence spread > CDH-SHRINK's influence spread > CDH-Kcut's influence spread > DH's influence spread. Since EGA is too time consuming, we only show the influence spread from 5 seeds to 30 seeds. When the number of seeds is larger than 33, EGA will never terminate on our computer.

Figures 12(a) and 12(b) depict the influence spread of 10 seeds and 30 seeds with different activation threshold (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. Similar to the NETHep dataset, when 30 seeds are selected, CIM's influence spread $\approx$ EGA's influence spread > CDH-SHRINK's influence spread > CDH-Kcut's influence spread > DH's influence spread. Furthermore, the influence of CIM is better than that of EGA in the cases $\theta = 0.3$ and 0.4. This is an interesting result; we find that under the criteria of a larger number of selected seeds and larger activation threshold, we can derive a better community structure. On the contrary, in Figure 12(a), we can also see that DH's influence spread is better than that in Figure 12(b). This is partly because the larger number of seeds may increase the possibility of information overlapping. The top-degree nodes usually all stay in some large community.



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

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

Fig. 13. Influence spread of different algorithms on Facebook with different flow durations.

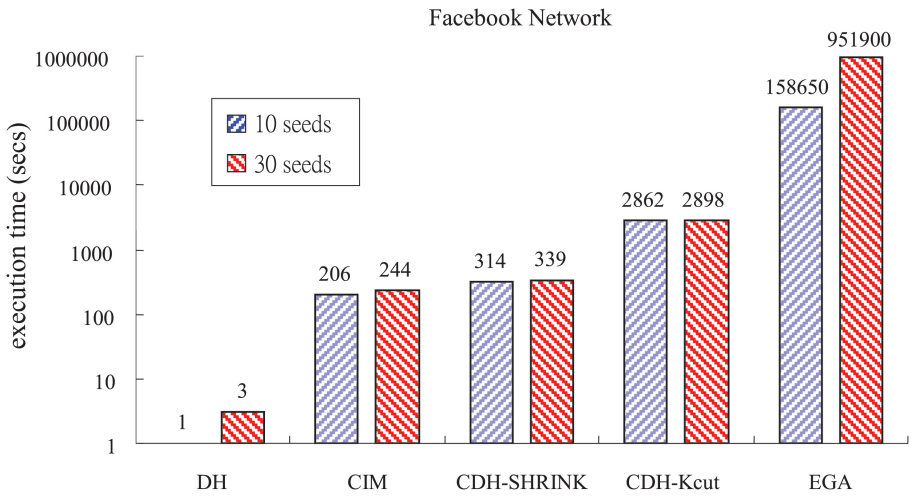


Fig. 14. Running time of different algorithms on the Facebook network when selecting 10 seeds and 30 seeds, respectively.

Figures 13(a) and 13(b) illustrate the influence spread of 10 seeds and 30 seeds with different flow duration (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 Figure 13(a), CIM's influence spread > 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 Figure 13(b), the ranking of influence spread becomes CIM $\approx$ CDH-SHRINK $\approx$ EGA > CDH-Kcut > DH. EGA still has the most influence spread in some cases. It is noticed that no matter what parameter values are set on FB, CIM's influence spreads are close to those of EGA, or even better in some cases.

We present the experiment on execution times of the five algorithms with 5 seeds and 30 seeds in Figure 14. The x-axis indicates the different algorithms, and the y-axis (logarithmic scale) indicates the execution time. CIM has the most efficient execution time. Just like on NETHep, we can observe that the running time of EGA is proportional to the number of seeds. Overall, in terms of influence spread,

CIM $\approx$ EGA $>$ CDH-SHRINK $>$ CDH-Kcut $>$ DH; in terms of efficiency, DH $>$ CIM $>$ CDH-SHRINK $>$ CDH-Kcut $>>$ EGA. One point which deserves mentioning is that due to the phenomenon of seed clustering, CIM will get better performance in terms of influence spread with high activation and with a larger number of required seed nodes.

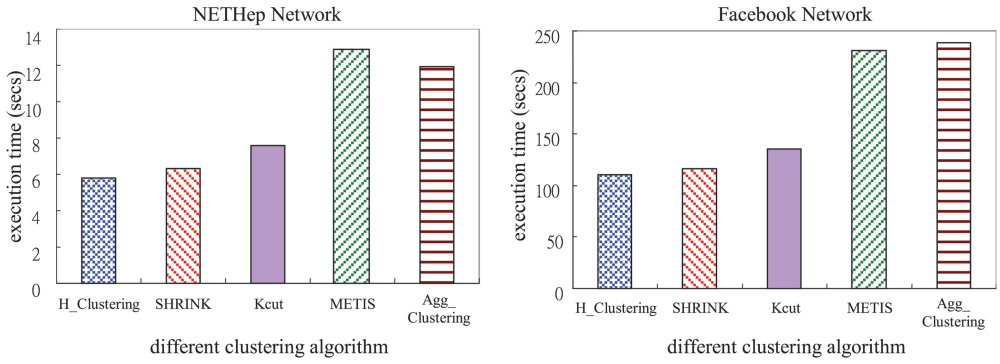
4.3. The Performance Comparison with Existing Clustering Algorithms

Given a social network, phase (i) of CIM aims to discover the community structure of the network to facilitate seed selection. Our proposed H_Clustering not only discovers the communities efficiently but also detects the hub nodes connecting to many communities. To show the effectiveness and the efficiency of the proposed H_Clustering, we detail some existing clustering algorithms, including two agglomerative clustering algorithms [Girvan and Newman 2002; Newman 2004], METIS [Karypis and Kumar 1996], Kcut [Ruan and Zhang 2007], and SHRINK [Huang et al. 2010].

The first agglomerative clustering algorithm [Girvan and Newman 2002] uses betweenness centrality [Freeman 1977] to evaluate the distance of two adjacent nodes and then removes edges with high betweenness progressively in each iteration to reveal the underlying community structure of the network. The second agglomerative method [Newman 2004] directly chooses the join (i.e., merging pair of nodes or communities together) that results in the greatest increase (or smallest decrease) in modularity at each step to hierarchically find the community structure without evaluating the similarity of each edge. METIS [Karypis and Kumar 1996] presents a parallel multilevel network partitioning algorithm. It first reduces the size of the network by collapsing vertices and edges, then roughly partitions the smaller network into several parts, and finally constructs the partition for the original network by projecting and refining the partitions to successively finer network. Kcut [Ruan and Zhang 2007] combines the recursive partitioning and direct k-way method based on the eigenvectors of the Laplacian matrix of a network. Note that, for these four clustering algorithms, every node in the network belongs to only one community, that is, the communities do not overlap. SHRINK [Huang et al. 2010] combines the advantages of the density-based clustering and modularity optimization methods. It uses cosine similarity to calculate the structural similarity of two adjacent nodes and then hierarchically finds the community structure. SHRINK adopts the concept of hub nodes in a network (i.e., nodes connecting to several communities) when finding community structures.

To demonstrate the strength of using H_Clustering in the CIM framework, we implement several clustering algorithms, including METIS [Karypis and Kumar 1996], Kcut [Ruan and Zhang 2007], and SHRINK [Huang et al. 2010], and one agglomerative clustering algorithm for performance comparison. Note that compared to the clustering algorithms, that is, METIS, Kcut, and SHRINK, the two agglomerative clustering algorithms presented in Girvan and Newman [2002] and Newman [2004] have the worst execution times. In summary, the computation cost of betweenness [Girvan and Newman 2002] and the enumeration for the best join [Newman 2004] are very time consuming. Thus, we design another agglomerative clustering algorithm, called Agg_Clustering, for further performance comparisons. Specifically, Agg_Clustering uses *cosine similarity* as the similarity measure of two nodes. The same as the aforementioned agglomerative clustering algorithms [Girvan and Newman 2002; Newman 2004], Agg_Clustering merges two nodes (or communities) into one community in each iteration, if they have the largest similarity in the network. The result is a hierarchical tree. Finally, we cut the tree at the level that has the largest modularity to produce the community structure.

We first discuss the efficiency of the proposed H_Clustering algorithm. All experiments are conducted using two real datasets: NETHep and Facebook. Figure 15 shows the result in terms of execution time. From the figures, we can observe that



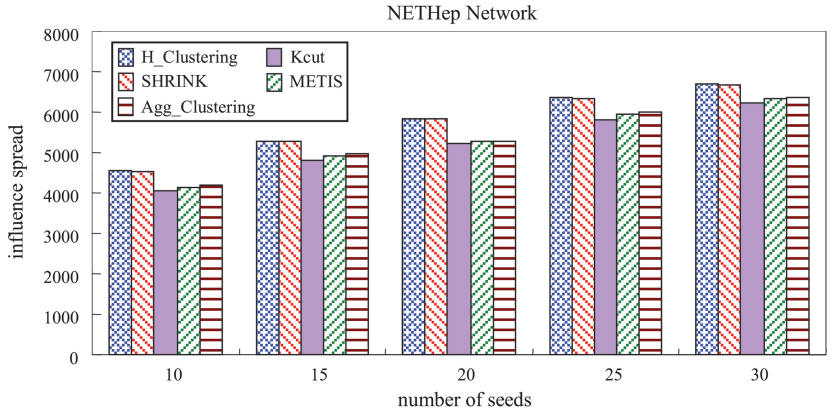
(a) Runtime on NETHep dataset.

(b) Runtime on Facebook dataset.

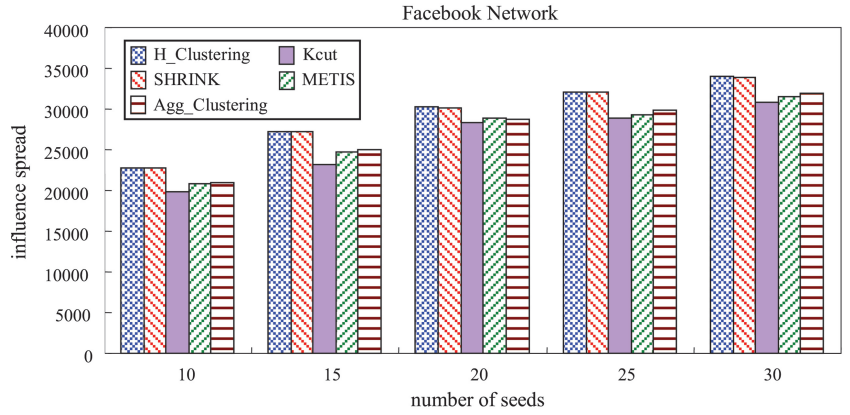
Fig. 15. The runtime performance of different clustering algorithm on real datasets.

H_Clustering incurs the lowest execution time for discovering the community structure. This is mainly because H_Clustering stops when the clustering result is good enough, without continuing to force naturally homeless nodes (hubs and outliers) into communities. The hub nodes discovered by H_Clustering provide critical position information valuable for seed selection in the CIM framework. Although SHRINK can also discover hub nodes in the network, it spends more time deriving the scores of structure similarity between nodes. Additionally, instead of only merging the pair of nodes with the largest similarity, H_Clustering groups each pair of nodes into a community if the structural similarity between these two nodes is the largest among their surrounding edges from each other. For example, given two nodes u and v, suppose the edge (u, v) is the largest among all edges connecting to u and is also the largest among all edges connecting to v, but not the largest edge in the network. Agg_Clustering will not merge u and v into a community. However, H_Clustering can merge u and v into a community. Obviously, the merge strategy of H_Clustering can efficiently reduce the processing iteration of clustering and further improve the runtime performance. This strategy is very important for hierarchical clustering. From the experiments, H_Clustering only requires 25 and 123 iterations to find the community structure in the NETHep and Facebook datasets, respectively. However, Agg_Clustering needs 193 and 1,107 iterations to discover the community results. This is the main reason why the execution time of H_Clustering outperforms that of Agg_Clustering. As shown in Figure 15, the execution times of H_Clustering are 82% and 143% faster than those of Agg_Clustering on the NETHep and Facebook datasets, respectively.

Next, we discuss the effectiveness of the proposed H_Clustering algorithm. Notice that in this article, we are mainly interested in the problem of influence maximization instead of which clustering algorithm has a better result or precision for community structure discovering. In other words, we intend to find a clustering algorithm applicable for the CIM framework to efficiently and effectively address the influence maximization problem. Now we investigate how the community structures discovered from different clustering algorithms may affect the influence spread in the CIM framework. We use the discovered results of different algorithms as the output of phase (i) in CIM (the input of phase (ii) of CIM). Figure 16 depicts the influence spread obtained by using the clustering results of different clustering algorithms on the NETHep and Facebook datasets, respectively. By increasing the number of seeds from 10 to 30, Figure 16 shows the information spread obtained with different clustering algorithms. It is easy to observe that utilizing H_Clustering and SHRINK can obtain more influence



(a) Influence spread on the NETHep dataset with different clustering algorithm in CIM framework.



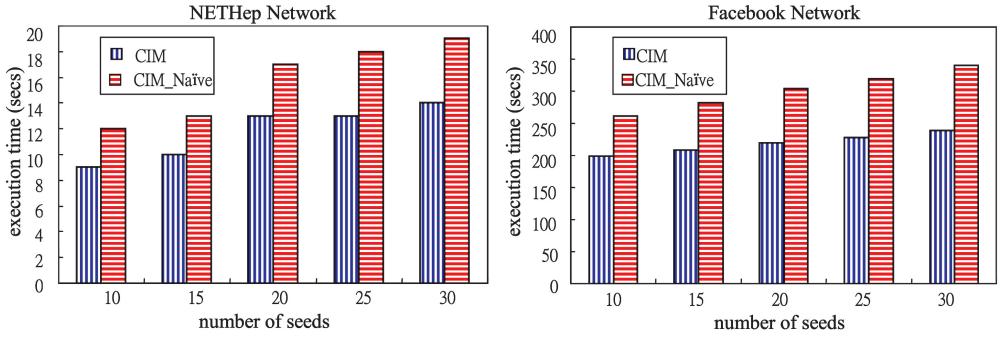
(b) Influence spread on the Facebook dataset with different clustering algorithm in CIM framework.

Fig. 16. The affect of influence spread with different clustering algorithm utilized in CIM.

spread over all the other clustering algorithms, owing to the hub nodes discovered from H_Clustering and SHRINK. Nevertheless, it is worth noting that, while H_Clustering and SHRINK algorithms obtain similar influence spread, H_Clustering is about 7%-8% more efficient than SHRINK, as shown in Figure 15. The experiments show that hub nodes can offer more opportunities to spread influence to other nodes. Furthermore, the performance of H_Clustering is good compared to that of other existing clustering algorithms. In summary, H_Clustering algorithm is a good choice for phase (i) in the CIM framework.

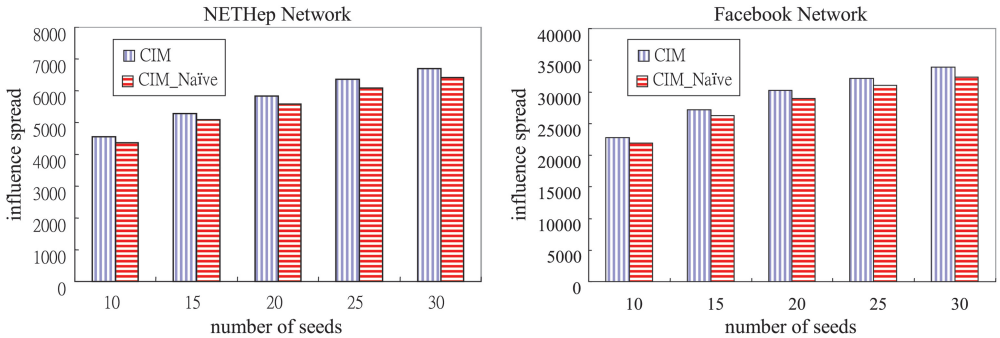
4.4. The Impact of Selecting Top Significant Communities and Seed Quota-Allocation

Phase (ii) of CIM is to generate candidate seeds. In Section 3.2, we have mentioned that one naïve approach is to select the centroid nodes of the k-largest communities in the social network as the k-influential seeds. In phase (ii) of CIM, we judiciously determine the top significant communities and allocate seed quotas for them. To investigate the impact of our design in phase (ii) of CIM, we implemented this naïve approach, denoted as CIM_Naïve. All experiments are conducted on the two real datasets: NETHep and



(a) Runtime on the NETHep dataset. (b) Runtime on the Facebook dataset.

Fig. 17. The affect of runtime performance of improvement strategy for CIM.



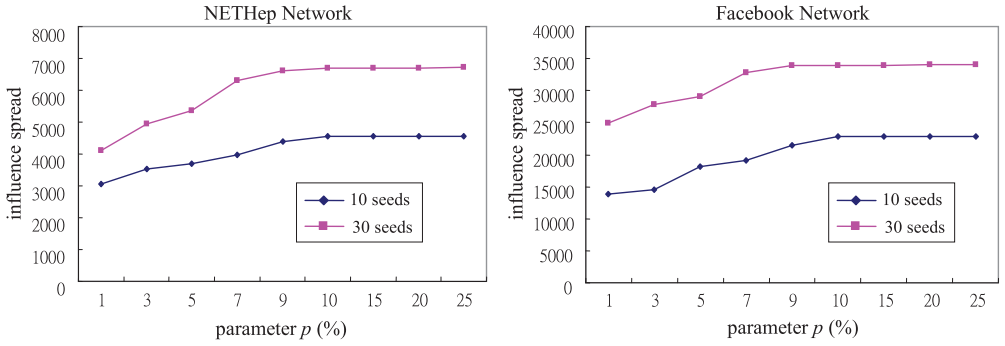
(a) Influence spread on the NETHep dataset. (b) Influence spread on the Facebook dataset.

Fig. 18. The affect of influence spread of improvement strategy for CIM.

Facebook. Figures 17 and 18 are the results of varying the number of selected seeds from 5 to 30.

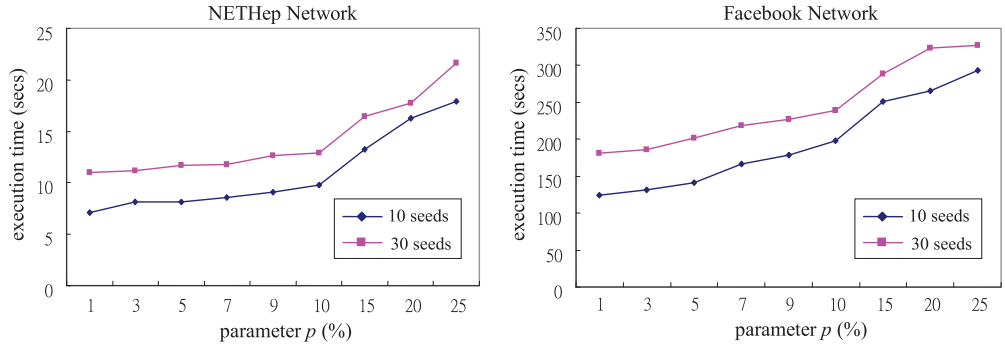
As shown in Figures 17(a) and 17(b), CIM is able to reduce about 28.4% (respectively, 37.9%) execution time compared with CIM_Naïve in NETHep dataset (respectively, Facebook dataset). Both top significant communities and seed quota-allocation are able to reduce the search space, thereby reducing runtime. Figures 18(a) and 18(b) show the impact of the top significant communities and seed quota-allocation in CIM on the influence spread. As can be seen in Figure 18, the influence spread of CIM_Naïve is about 4.1% and 4.6% smaller than CIM in the NETHep dataset and Facebook dataset, respectively. This is due to the fact that seeds in larger communities can have more influence spreads. These experiments justify our observations in Section 2.2. In other words, the size of community is a dominant factor for seed placement. The seed quota-allocation strategy allocates appropriate numbers of seeds for each significant community.

Now we investigate the impact of the setting of parameter p on the results of candidate generation. As mentioned in Section 3.2, CIM collects the top-p% of high-degree nodes and large-score-sum nodes in each significant community of the candidate set. The setting of parameter p is critical for the runtime performance and the result of influence spread of CIM. If p is too large, the candidate set will also become too large and may include many unqualified nodes for seed selection and thus increase the execution time of CIM. Nevertheless, if p is too small, the candidate set also will become too



(a) Influence spread on the NETHep dataset. (b) Influence spread on the Facebook dataset.

Fig. 19. The influence spread of CIM with varying top-p% nodes in the candidate set.



(a) Runtime on the NETHep dataset. (b) Runtime on the Facebook dataset.

Fig. 20. The runtime performance of CIM with varying top-p% nodes in the candidate set.

small and may not consist of a sufficient number of qualified nodes for seed selection, thereby decreasing the influence spread of CIM. Figures 19 and 20 depict the influence spread and the execution time with varying parameter p on the NETHep and Facebook datasets of 10 and 30 seeds, respectively. From the observation, as in Figure 19, when p is larger than about 10%, the influence spread will not increase with enlarging p. However, on the contrary, as in Figure 20, the execution time is increased when p is larger than about 10%. Consequently, we can conclude that, in most cases, including the top-10% of nodes in each significant community in the candidate set is sufficient to select good seed nodes. Hence, CIM collects the top-10% of nodes in each significant community to build the candidate set.

5. CONCLUSIONS

In this article, we present a framework, CIM, based on a heat diffusion model by integrating the information of community structure and the seed allocation method. The goal of this work is to address the efficiency issue of influence maximization problem while not compromising the quality of influence spread obtained based on the heat diffusion model. Due to the increasing scale of social networks nowadays, the requirement of efficiency on mining algorithms is critical for many applications. We develop a hierarchical clustering algorithm, H_Clustering, to detect the community structure and hubs of a social graph. CIM utilizes the community structure to effectively reduce

the overlapping information. We propose two optimization strategies, quota-allocation and insignificant pruning, to enhance the influence spread and to speed up the execution time of CIM according to community size. To the best of our knowledge, CIM is the first algorithm for utilizing community structure to solve the influence maximization problem. The experimental results on real-world and synthetic datasets show that our proposed CIM achieves superior performance in terms of running time and influence spread. As the next step, to better capture the real-world phenomenon, we plan to consider the influence maximization problem under the context of 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 worthy investigation to explore dynamic community structures to solve the influence maximization problem.

REFERENCES

R. Albert, H. Jeong, and A. Barabasi. 1999. Diameter of the World Wide Web. *Nature* 401, 130–131.

A. Barabasi and R. Albert. 1999. Emergence of scaling in random networks. *Science* 286, 509–512.

D. Bortner and J. Han. 2010. Progressive clustering of networks using structure-connected order of traversal. In *Proceedings of the 26th IEEE International Conference on Data Engineering (ICDE'10)*. 653–656.

Y. Chen, S. Chang, C. Chou, W. Peng, and S. Lee. 2012. Exploring community structures for influence maximization in social networks. In *Proceedings of the 6th SNA-KDD Workshop on Social Network Mining and Analysis held in conjunction with KDD'12 (SNA-KDD'12)*. 1–6.

W. Chen, Y. Wang, and S. Yang. 2009. Efficient influence maximization in social networks. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'09)*. 199–208.

L. Danon, J. Duch, A. Diaz-Guilera, and A. Arenas. 2005. Comparing community structure identification. *J. Stat. Mech. Theory Exp. 2005,* 9.

P. Domingos. 2005. Mining social networks for viral marketing. *IEEE Intell. Syst.* 20, 1, 80–93.

P. Domingos and M. Richardson. 2001. Mining the network value of customers. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'01)*. 57–66.

P. Domingos and M. Richardson. 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)*. 61–70.

M. Ester, H. Kriegel, J. Sander, and X. Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD'96)*. 226–231.

P. Estevez, P. Vera, and K. Saito. 2007. Selecting the most influential nodes in social network. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN'07)*. 2397–2402.

Z. Feng, X. Xu, N. Yuruk, and T. Schweiger. 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)*. 385–396.

L. Freeman. 1977. A set of measures of centrality based on betweenness. *Sociometry* 40, 1, 35–41.

M. Girvan and M. Newman. 2002. Community structure in social and biological networks. *Proc. Nat. Acad. Sci.* 99, 12, 7821–7826.

J. Goldenberg, B. Libai, and E. Muller. 2001. Talk of network: A complex systems look at the underlying process of word-of-mouth. *Market. Lett.* 12, 3, 211–223.

M. Gomez-Rodrigues, D. Balduzzi, and B. Scholkopf. 2011. Uncovering the temporal dynamics of diffusion networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML'11)*. 561–568.

M. Gomez-Rodriguez, J. Leskovec, and A. Krause. 2010. Inferring networks of diffusion and influence. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining (KDD'10)*. 1019–1028.

J. Huang, H. Sun, J. Han, H. Deng, Y. Sun, and Y. Liu. 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)*. 219–228.

G. Karypis, and V. Kumar. 1996. Parallel multilevel k-way partitioning scheme for irregular graphs. In *Proceedings of the ACM/IEEE Conference on Supercomputing (SC'96)*. 1–21.

D. Kempe, J. Kleinberg, and E. Tardos. 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)*. 137–146.

A. Lancichinetti, S. Fortnato, and J. Keryesz. 2009. Detecting the overlapping and hierarchical community structure in complex network. *New J. Physics* 11, 3.

A. Lancichinetti, S. Fortnato, and F. Radicchi. 2008. Benchmark graphs for testing community detection algorithms. *Phy. Rev. E* 78, 4.

H. Ma, H. Yang, M. Lyu, and I. King. 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)*. 233–242.

S. Myers and J. Leskovec. 2010. On the convexity of latent social network inference. In *Proceedings of the 24th Annual Conference on Neural Information Processing Systems (NIPS'10)*. 1741–1749.

A. Ng, M. Jordan, and Y. Weiss. 2001. On spectral clustering: Analysis and an algorithm. In *Proceedings of the Conference on Neural Information Processing Systems: Natural and Synthetic (NIPS'01)*. 849–856.

M. Newman. 2004. Fast algorithm for detecting community structure in networks. *Phy. Rev. E* 69, 6.

M. Newman. 2006. Modularity and community structure in networks. *Proc. Nat. Acad. Sci. U.S.A.* 103, 23, 8577–8582.

G. Palla, I. Derenyi, I. Farkas, and T. Vicsek. 2005. Uncovering the overlapping community structure of complex networks in nature and society. *Nature* 435. 814–818.

E. ROGERS. 2003. *Diffusion of Innovations*. 5th Ed., Free Press, New York, NY.

J. Ruan and W. Zhang. 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)*. 643–648.

K. Saito, M. Kimura, K. Ohara, and H. Motoda. 2011. Efficient discovery of influential nodes for SIS models in social networks. *Knowl. Inform. Syst.* 30, 3, 613–635.

T. Valente. 1995. *Network Models of the Diffusion of Innovations*. Hampton Press.

L. Wan, J. Liao, and X. Zhu. 2008. Finding evaluating community structure in social networks. In *Proceedings of the 4th International Conference on Advanced Data Mining and Applications (ADMA'08)*. 620–627.

Y. Wang and X. Feng. 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)*. 350–361.

S. Wasserman and K. Faust. 1994. Social network analysis: Methods and applications. Cambridge University Press.

S. White and P. Smyth. 2005. A spectral clustering approach to finding communities in graph. In *Proceedings of the 5th SIAM International Conference on Data Mining (SDM'05)*. 274–286.

X. Xu, N. Yuruk, Z. Feng, and T. Schweiger. 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)*. 824–833.

H. Young. 2000. The diffusion of innovations in social networks. Economics Working Paper 437, Johns Hopkins University.

W. Zachary. 1997. An information flow model for conflict and fission in small group. *J. Anthro. Res.* 33, 452–473.

Received November 2012; revised February, May 2013; accepted June 2013