Knowledge-Based Systems 117 (2017) 88–100

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

CoFIM: A community-based framework for influence maximization on large-scale networks

Jiaxing Shang[a,b,*], Shangbo Zhou[a,b], Xin Li[c], Lianchen Liu[d], Hongchun Wu[a,b]

[a] College of Computer Science, Chongqing University, Chongqing 400044, China
[b] Key Laboratory of Dependable Service Computing in Cyber Physical Society, Ministry of Education, Chongqing University, Chongqing 400044, China
[c] Department of Information Systems, College of Business, City University of Hong Kong, Hong Kong SAR, China
[d] Department of Automation, Tsinghua University, Beijing 100084, China

ARTICLE INFO

Article history:
Received 30 March 2016
Revised 17 July 2016
Accepted 28 September 2016
Available online 30 September 2016

Keywords:
Influence maximization
Community structure
Large-scale networks
Diffusion model
Computational complexity

ABSTRACT

Influence maximization is a classic optimization problem studied in the area of social network analysis and viral marketing. Given a network, it is defined as the problem of finding k seed nodes so that the influence spread of the network can be optimized. Kempe et al. have proved that this problem is NP hard and the objective function is submodular, based on which a greedy algorithm was proposed to give a near-optimal solution. However, this simple greedy algorithm is time consuming, which limits its application on large-scale networks. Heuristic algorithms generally cannot provide any performance guarantee. To solve this problem, in this paper we propose CoFIM, a community-based framework for influence maximization on large-scale networks. In our framework the influence propagation process is divided into two phases: (i) seeds expansion; and (ii) intra-community propagation. The first phase is the expansion of seed nodes among different communities at the beginning of diffusion. The second phase is the influence propagation within communities which are independent of each other. Based on the framework, we derive a simple evaluation form of the total influence spread which is submodular and can be efficiently computed. Then we further propose a fast algorithm to select the seed nodes.

Experimental results on synthetic and nine real-world large datasets including networks with millions of nodes and hundreds of millions of edges show that our algorithm achieves competitive results in influence spread as compared with state-of-the-art algorithms and it is much more efficient in terms of both time and memory usage.

1. Introduction

Influence maximization (IM) is a classic network optimization problem. It orients from the area of viral marketing [1,2]. Consider that a company has developed a new product and wants to promote it among the customers. The company plains to select some people and let them try this product for free, hoping that these people could recommend it to their family and friends. As an expectation, the product will be largely adopted in the network with the "world-of-mouth" effect [3]. However, most of the time the budget of a company is limited, then a natural question is: how to select the "best" initial adopters so that the product can be most widely adopted? Mathematically speaking, it is defined as the problem of finding k seed nodes in a network such that they cause the maximum scale of cascading, which we call influence maximization.

1.1. Diffusion model

The study of influence maximization relies on the diffusion models [4]. Currently the most widely used models are IC (Independent Cascade) model and LT (Linear Threshold) model [5]. In the two models, each node is in one of two states: *active* or *inactive*. Active nodes are those who have adopted the product and will propagate it to their neighbors. Inactive nodes are those who have not heard of the product or rejected to adopt it. Initially all nodes are inactive, then k seed nodes are selected to be activated and propagation starts from the k nodes.

IC model: In this model, at step t, for an active node u, it will try to activate each of its inactive neighbor v, and succeed with probability p_{uv}. Node u has only one chance to activate v, whether succeed or not, u will make no attempt to activate v in the fu-

* Corresponding author.
E-mail address: shangjx@cqu.edu.cn (J. Shang).

http://dx.doi.org/10.1016/j.knosys.2016.09.029
0950-7051/

ture. If v was successfully activated, then from step $t+1$, v will be active and try to activate its inactive neighbors. If no node is activated as step T, the diffusion process will stop.

LT model: In this model, the activation of node v depends on the set of its active neighbors. For each directed edge from u to v, there is a weight $b_{u,v}$ indicating the influence of u on v. For each node v, the constraint $\Sigma_{u\,\text{neighbor of}\,v} b_{uv} \leq 1$ must be satisfied. Node v has an activation threshold θ_v, which is between 0 and 1. Once the condition $\Sigma_{u\,\text{neighbor of}\,v} b_{uv} \geq \theta_v$ is satisfied, v will become active. When no more nodes can be activated, the diffusion process will stop.

1.2. Influence maximization

The influence maximization problem was firstly formatted by Kempe et al. [5]. Given a network $G(V, E)$, influence maximization aims to find a subset S of $k = |S|$ vertices such that the diffusion orients from S can cause the maximum cascade of influence, i.e.,

$$S^* = \arg_S \max \sigma(S) \tag{1}$$

where $\sigma(S)$ is an objective function evaluating the influence spread, which is defined as the expected number of active users in the network after the difussion stopped.

Kempe et al. [5] proved that under traditional linear threshold (LT) and independent cascade (IC) diffusion models, this problem is NP hard, and the objective function is submodular. For arbitrary function $\sigma(\cdot)$ that maps subsets of a finite ground set U to non-negative real numbers, we say the function is submodular, if it satisfies the following two properties: (1) monotone increasing, i.e., $\forall S \subset T$, $\sigma(S) \leq \sigma(T)$; (2) diminishing returns, i.e., $\forall v \in V$, $S \subseteq T$, we have $\sigma(S \cup \{v\}) - \sigma(S) \geq \sigma(T \cup \{v\}) - \sigma(T)$, which means that the marginal gain from adding an element to a set S is at least as high as the marginal gain from adding the same element to a superset of S. Based on the mathematical properties of submodular funcitons [6], Kempe et al. proposed a "hill-climbing" greedy algorithm to solve this problem, which begin with an empty set of S, and then iteratively add an element to S that brings the maximum marginal gain, util $|S| = k$. Kempe et al. proved that the solution provided by the greedy algorithm provides a factor of $(1 - 1/e - \varepsilon)$ performance guarantee to the optimal solution under both the LT and IC models. Here e is the base of the natural logarithm and ε is any positive real number. The second item $1/e$ orients from the submodular function itself. The third item ε is the error of approximating the objective function using Monte–Carlo simulations. Theoretically, the traditional greedy algorithm provides a 63% guarantee to the optimal solution. In real experiments, the solution provided by the greedy algorithm is quite close to the optimal solution. However, to have a good approximation of the objection function given the seed set S, the greedy algorithm requires tens of thousands of Monte–Carlo simulations, which seriously limits its application on large-scale networks.

To solve the time efficiency problem of traditional greedy algorithm, a spectral of algorithms were proposed by researchers in recent years. Some works make use of submodularity, such as the CELF algorithm proposed by Leskovec et al. [7]. Some research works assume that the influence can spread on the network only through shortest paths [8–10] so that the objective function can be exactly computed. Another way to reduce the time complexity is to simply select top k nodes based on some heuristic metrics [11–13], such as the degree centrally, betweeness centrality et al. However, since the heuristic methods take no consideration of propagation models, they usually give poor solutions.

In the age of big data, network scale grows in millions, if not billions. Traditional influence maximization methods either cannot handle large-scale networks, or provide inaccurate solutions with low influence spread. To solve the problem of traditional influence maximization algorithms, in this paper we propose CoFIM: a **Co**mmunity-based **F**ramework for **I**nfluence **M**aximization on large-scale networks. In our framework the influence propagation process is divided into two phases: (i) seeds expansion; and (2) intra-community propagation. The first phase is the expansion of seed nodes among different communities at the beginning of diffusion. The second phase is the influence propagation within communities which are independent of each other. Based on the framework, we derive a simplified evaluation form of the total influence spread which is sumodular and can be efficiently computed. Then we further develop a fast greedy algorithm to select the seed nodes.

Experimental results on synthetic and nine real-world large datasets including networks with millions of nodes and hundreds of millions of edges show that our algorithm can significantly outperform other state-of-the-art methods in terms of time and memory efficiency with almost no comprise on accuracy as evaluated by the influence spread.

The rest of this paper is organized as follows. Section 2 reviews the literature on influence maximization. Section 3 elaborates the preliminaries of the problem to be addressed. Section 4 introduces our CoFIM framework. Section 5 presents the evaluation framework, including the datasets, evaluation metrics, baseline methods, and experimental procedure. Experiment results are presented in Section 6. Section 7 concludes this paper.

2. Literature review

Since Kempe et al. [5] formally formatted the influence maximization problem and proposed the greedy algorithm, a lot of research works have been published to tackle this problem. Generally, these works can be divided into four categories: (1) submodularity-based algorithms; (2) centrality-based heuristic algorithms; (3) influence path based-algorithms; and (4) community-based algorithms. Here we give a brief review on these research works.

2.1. Submodularity based algorithms

Svirdenko et al. [14] extended the problem defined by Kempe et al. [5] by considering node prices. Unlike the original problem which simply selects k seed nodes with unit price, their model assumed that nodes are selected with different prices and added a constraint condition: $\sum_i c_i = B$, where c_i is the price of selecting node i and B is the total budget. It is easy to see that when $c_i = 1, \forall i \in V$, this problem reduces to the traditional influence maximization problem. Svirdenko et al. proved that under the IC and LT models the objective function of the problem is still submodular, meaning that the hill-climbing greedy algorithms can be applied and a performance guarantee to the optimal solution can be achieved.

Leskovec et al. [7] made further use of submodularity property to improve the time efficiency of the greedy algorithm and proposed the CELF (Cost-Effective Lazy Forward) algorithm. Specifically, during each iteration of the greedy algorithm, it maintains the marginal gain of each node. In the next round of iteration, if the current marginal gain of a node is higher than the marginal gain of others nodes in the previous round of iteration, then from the "diminishing returns" property of the submodular function, we know the current marginal gain of this node must be higher than the marginal gain of other nodes in the current round of iteration (the marginal of any node in the current round of iteration must be smaller than or equal to its marginal gain in the previous round of iteration). This means there will be no need to evaluate the current marginal gain of others nodes, which significantly reduces the time complexity. Experimental results shows that on some datasets

the CELF algorithm is 700 times faster than the simple greedy algorithm. At the same time, the CELF algorithm outputs the same result as the simple greedy algorithm.

Inspired by similar idea of the CELF algorithm, Goyal et al. [15] proposed CELF++, an enhanced version of the CELF algorithm. When computing the marginal gain of node u with respect to the current seed set S, the CELF++ algorithm will simultaneously compute the marginal gain of u with respect to $S \cup \{v\}$, where v is the node that brings the maximum marginal gain with respect to S in the current iteration by now. Since the two values of marginal gain can be computed simultaneously in one round of Monte–Carlo simulation while the CELF algorithms needs two rounds of Monte-Carlo simulations, the time efficiency can be improved. Experimental results show that the CELF++ algorithms is about 30% $\sim$ 50% faster than the CELF algorithm.

Although current submodularity-based algorithms are much faster than the simple greedy algorithm, they still cannot handle large-scale networks since all these algorithms have to perform the time consuming Monte-Carlo simulations to approximate the influence spread of a given seed set S.

2.2. Centrality based heuristic algorithms

To get the "best" k seed nodes corresponding to the maximum influence spread, a simple idea is to select the top k nodes based on some predefined centrality metrics (e.g., degree centrality, betweenness centrality, et al.). However, previous study showed that simply selecting top k nodes based on degree centrality may produce inaccurate results. In order to solve this problem, some research works proposed new metrics which are more accurate.

In [11] Chen et al. proposed an algorithm named SD (Single Discount) by considering the effect of already selected nodes on current candidates. Initially, all nodes are sorted based on degree, then the algorithm iteratively selects a node with the maximum degree and adds it to the seed set S. Once a node is selected, for each of its neighbor, the neighbor's degree will be reduced (discounted) by 1. The intuition behind this algorithm is that after a node is selected, it will no longer be influenced by its neighbors. So for each of its neighbor, the number of nodes the neighbor can influence will be reduced by 1. Experimental results show that the SD algorithm outperforms the degree based algorithm in terms of influence spread. However, since SD only uses the network topology without consideration of the specific diffusion model, its improvement is very limited.

Wang et al. [12] proposed an algorithm named TW (Targeted Wise) based on node potential, where the potential of a node u is defined as:

$$p(u) = \sum_{v \in N(u), v \notin A(u)} b_{uv} \tag{2}$$

where $N(u)$ is the neighbors of u, $A(u)$ is the active neighbors of u, and b_{uv} is the influence probability of u on v. The selection of seed nodes is divided into two steps and is manipulated by a parameter c. In the first step, $k_1 = (1-c)k$ nodes with the highest potential values are selected. In the second step, the rest $k_2 = k - k_1 = c \cdot k$ nodes are selected using traditional greedy algorithm. It is easy to see that when $c = 1$, the algorithm reduces to the greedy algorithm. Though this algorithm is faster than the greedy algorithm, its disadvantages are also obvious. First, the selection of parameter c can only based on experience. Second, the algorithm contains the simple greedy part (when $c \neq 0$), meaning that its time efficiency cannot be fundamentally improved.

Kundu et al. [13] proposed an algorithm that selects top k nodes based on their diffusion degree. The diffusion degree of a node v is defined as:

$$C_{DD}(v) = \lambda_v \cdot C_D(v) + \sum_{i \in N(v)} \lambda_i \cdot C_D(i) \tag{3}$$

where $C_D(v)$ is the degree centrality of v. For undirected networks, it is equal to v's degree k_v. λ_v is the influence probability of v, i.e., when v is active, it will try to activate each of its inactive neighbor and succeed with probability λ_v. From the definition we see that diffusion degree contains two parts: (i) the diffusion from v to its neighbors, and (ii) the diffusion from v's neighbors to the rest of network. The algorithm selects top k nodes with the highest diffusion degree as the seed set S. Compared to the simple degree based algorithm, this algorithm considers two steps of diffusion, so it can generate more accurate solutions.

2.3. Influence path based algorithms

Influence path-based algorithms generally assume that influence can spread along the network only through some special paths, e.g., shortest paths or paths with length smaller than a given threshold. Based on this assumption, the objective function, i.e., influence spread can be efficiently approximated without Monte-Carlo simulations.

Kimura et al. [8] firstly proposed SP1M (Shortest Path 1 Model), a shortest path-based influence maximization algorithm. They made this assumption that only the shortest and second shortest paths can spread influence. Based on this model, they showed that the influence spread can be recursively computed, which avoids the long time Monte-Carlo simulations. Obviously, this algorithm takes an approximation strategy to the objective function to improve its time efficiency.

Inspired by similar idea, Chen et al. [9] proposed MIA (Maximum Influence Arborescence), an algorithm which uses local graph structure — arborescence — to approximate the influence spread. In this algorithm each path has a propagation probability, defined as the product of propagation probability of edges along the path. The authors assumed that influence can spread only through paths with the maximum propagation probability. An influence path is defined as the path with the maximum propagation probability. The arborescence of node u is defined as the set of nodes located in the paths whose influence probability is higher than a threshold θ and orient from u. Based on the local arborescence structure the influence of any given seed set S can be effectively approximated. Moreover, Chen el al. also proved that for the classic influence maximization problem, computing the exact value of influence spread under the IC and LT models is #P hard.

Kim et al. [10] proposed another influence path-based algorithm IPA (Independent Path Algorithm) which made the assumption that different influence paths were independent of each other. Unlike Chen et al.'s work which only considered shortest influence paths, Kim et al.'s algorithm considered all paths as long as their influence probability was larger that a given threshold θ. Since these paths were assumed to be independent of each other, the influence spread can be efficiently computed in a parallel manner.

Compared with submodularity-based and centrality-based algorithms, influence path-based algorithm achieve a balance between efficiency and effectiveness. However, these algorithms also have some disadvantages, such as they can not provide any theoretical guarantee to the optimal solutions. As a result, on some datasets they perform much worse than greedy or submodualrity-based algorithms. Moreover, since these algorithms have to maintain a large amount of influence paths, they usually require huge memory.

2.4. Community-based algorithms

Recently, some research works were proposed to tackle the influence maximization problem using community information. Community structure [16] is defined as the partition of network nodes into groups, within which nodes are densely connected while between which they are sparsely connected. Community structure usually revels fundamental properties of networks [17,18] and the problem of community detection has been extensively studied in recent years [19–23]. Recent works also show that community structure plays an important role in epidemic spreading [24–26]. The general idea of community-based influence maximization algorithms is: based on the fact that different communities are sparsely connected to each other, we may use the influence of a node within its own community to approximate its influence on the whole network. Benefit from the fact that a community's size is usually much smaller than the whole network size, the influence of a node within its own community can be computed more efficiently.

Cao et al. [27] proposed the first community-based influence maximization algorithm OASNET (Optimal Allocation in a Social NETwork). They assume different communities are independent of each other and influence cannot spread across different communities. The community structure was detected by the CNM (Clauset–Newman–Moore) [28] algorithm. The selection of seed nodes contains two phrases. In the first phase, from each community the algorithm selects k nodes using traditional greedy algorithm, resulting in a total number of $C \cdot k$ candidate nodes. In the second phase the algorithm selects k nodes as the seed set S from the $C \cdot k$ candidates using dynamic programming.

Zhang et al. [29] studied the problem of identifying the influential nodes on networks with community structure. The authors firstly constructed an information transfer probability matrix from the weighted network. Then they applied the k-medoid clustering algorithm to identify the k seed (influential) nodes. They evaluated the performance of their algorithm on both LFR [30] synthetic networks and several real-world networks with ground truth community structure. Experimental results show that their algorithm can effectively find the most influential nodes especially on networks with unbalanced community structure. The algorithm even outperforms the simple greedy algorithm in terms of influence scope, a new evaluation metric defined by the authors. However, the effectiveness of this algorithms is only validated on small networks. It is not sure whether the algorithm can also handle large-scale networks with millions of nodes and edges.

Chen et al. [31] studied the community-based influence maximization problem using the HD (heat diffusion) model and proposed the CIM (Community-based Influence Maximization) algorithm. The algorithm can be divided into three phases: (i) community detection, (ii) candidate nodes generation, and (iii) seed nodes generation. In the first phase, the authors proposed a hierarchical community detection algorithm $H_{Clustering}$ to obtain the community structure. In the second phase, candidate nodes are selected based their network topology and community features. In the third phase, k nodes are selected from the candidates as the final seed set S based on some scoring metrics, e.g., whether the node is a hub, the number of communities connected by the node, et al. Experimental results show that this algorithm performs well under the heat diffusion model.

Li et al. [32] considered the node conformity and proposed the community-based influence maximization algorithm CINEMA (Conformity-aware INfluEnce MAximization). Based on conformity, the C^2 (Conformity-aware Cascade) diffusion model is defined. In the diffusion process, the probability that an active node u activates its inactive neighbor v is deinfed as: $p_{uv} = \Phi(u) \cdot \Omega(v)$, where $\Phi(u)$ is the influence index of u and $\Omega(v)$ is the conformity index of v. Similar to other community-based influence maximization algorithms, the algorithm firstly detects community structure by cutting edges so that different communities are isolated. The k seed nodes are selected from different communities, making sure that each new seed node brings the maximum marginal gain in influence spread. The authors also provided an upper bond between the algorithm's solution and the optimal solution.

From the above introduction, we see that community-based influence maximization algorithms are generally faster than traditional greedy algorithms. Moreover, since they usually make the assumption that different communities are isolated, these algorithms naturally support parallelization. However, the disadvantages of current community-based algorithms are also obvious. First, evaluating the marginal gain of a node within its own community also needs Monte-Carlo simulations, which is time consuming and limits the application of these algorithms on large-scale networks. Second, since the algorithms use the influence of a node within its own community to approximate its influence on the whole network, the algorithm's accuracy will severely rely on the underlying community structure. If the connections among different communities are very sparse, the approximation will be good. On contrary, the approximation will be bad and the algorithm may generate inaccurate results. Third, some community-based algorithms rely on specific diffusion model, it is not sure whether these algorithms will perform well under classis IC and LT models.

Besides the above introduced methods, there are also many other influence maximization methods, such as the voter model based algorithms [33,34], the graph pruning algorithms [35,36], the GPU accelerating algorithms [37], the competitive influence maximization algorithms [38,39], dynamic influence maximization [40], the time-restricted model based algorithm [41].

3. Preliminaries

3.1. Problem definition

The influence maximization problem was firstly formulated by Kempe et al. [5]. Under the IC or LT model, we use S to denote the seed set, i.e., the set of active nodes at step $t = 0$, and $S(t)$ as the set of active nodes at step t. It is easy to see that the propagation stops when $S(t) = \Phi$. Then the number of overall activated nodes after the propagation stopped can be represented by $\sum_{t=0}^{\infty} |S(t)|$. Since the diffusion models are usually stochastic, we use $\sigma(S)$ to denote the expected number of overall activated nodes. The influence maximization problem is defined as follows:

Definition 1. (Influence Maximization [5]) Given a network and a diffusion model, the influence maximization problem aims to find a subset S of k nodes ($|S| = k$), such that the expected number of overall activated nodes $\sigma(S)$ is maximized:

$$S^* = \arg_S \max \sigma(S) \tag{4}$$

Kempe et al. [5] proved that under IC and LT models, the influence maximization problem defined in Definition 1 is NP-hard and the objective function $\sigma(S)$ is submodular. A set function f: $2^V \rightarrow R$ is submodular if it has the following property:

$$f(S \cup v) - f(S) \geq f(T \cup v) - f(T), \forall v \in V, S \subseteq T \tag{5}$$

Based on the submodualrity, Kempe et al. further proposed a "hill-climbing" greedy algorithm (as shown in Algorithm 1) and proved that the algorithm provided a factor of $(1 - 1/e - \varepsilon)$ guarantee to the optimal solution, as shown in Theorem 1.

Theorem 1 (*Kempe et al. [5]*)**.** *For influence maximization problem defined in Definition 1, the seed set generate by Algorithm 1 provides*

Algorithm 1 The hill-climbing greedy algorithm (Kempe et al. [5]).

Require:
Network G, number of seed nodes k
Ensure:
Seed set S
1: Initialize: Let $S \leftarrow \Phi$.
2: **for** $i = 1$ to k **do**
3: $v = \arg_u \max\{\sigma(S \cup \{u\}) - \sigma(S)\}$
4: // $\sigma(*)$ is computed using Monte-Carlo simulations
5: $S \leftarrow S \cup \{v\}$
6: **end for**
7: **return** S

a factor of $(1 - 1/e - \varepsilon)$ *approximation to the optimal solution, i.e.,* $f(S) \geq (1 - 1/e - \varepsilon) f(S^*)$, *where* S^* *is the optimal solution.*

Theorem 2 (*Chen et al. [11]*). *The time complexity of Algorithm 1 is* $O(knRm)$, *where* k *is the number of seed nodes,* n *is the number of network nodes,* m *is the number of network edges, and* R *is the number of Monte–Carlo simulations in computing* $\sigma(S)$.

From Theorem 2 we see that the time complexity of the simple greedy algorithm proposed by Kempe et al. [5] is very high. In their experiments the parameter R is required to be as high as 10,000 to obtain a good approximation, which severely limits its application on large-scale networks.

3.2. Network preliminaries

Without loss of generality, we define an undirected unweighted graph[1] $G = (V, E)$, where V is the set of vertices and E is the set of edges. G has adjacency matrix A, where:

$$A_{i,j} = \begin{cases} 1 & \text{if } i \text{ and } j \text{ are connected} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

For any node v, its neighbor set is:

$$N(v) = \{u | A_{v,u} > 0\} \tag{7}$$

The community structure of G is defined as the partition of its nodes into subgroups $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$, where the ith community C_i contains multiple nodes (in this paper we only consider non-overlapping community structure, so there is no overlap across C_i). For any node v, let $C(v)$ denotes the community of v, i.e., $v \in C(v)$. Then we further have the neighbor community set of v as:

$$NC(v) = \{C(u) | u \in N(v)\} \tag{8}$$

Let S denotes any node set, then the neighbor set of S is:

$$N(S) = \bigcup_{v \in S} N(v) \tag{9}$$

Similarly, the neighbor community set of S is:

$$NC(S) = \{C(u) | u \in NC(S)\} \tag{10}$$

4. Community-based influence maximization framework

4.1. Two-phases diffusion model

Given any network with non-overlapping community structure, we assume that the diffusion can be divided into two phases: (i) seed expansion and (ii) intra-community propagation. The first phase is the expansion of seed nodes across different communities. The second phase is influence diffusion within each community and there is no inter-community propagation. We call seed set S as the first-order seeds, while the neighbor of S as second-order seeds. We assume the first phase is the propagation from S to $N(S)$ while the second phase is the propagation from $N(S)$ to the rest of network within communities.

Our model orients from the following facts: First, any diffusion starts from S must firstly pass $N(S)$, which is the seed expansion phase of our model (since nodes in $N(S)$ usually allocate in different communities, its effect is like expanding seeds to other communities). Second, from the property of community structure we know that nodes within communities are densely connected while between communities they are sparsely connected. Recent study [42] has shown that dense intra-community connections may trap the propagation within communities, which is the second phase of our model. Now we give detailed description of our model.

(1) Node expansion: This phase is the propagation from first-order seeds S to second-order seeds $N(S)$. Since the second order seeds may distribute in different communities, the seeds are expanded. Given the first-order seed set S, for any second-order seed node $v \in N(S)$, the probability that v is activated after first phase is:

$$\mathbb{P}_v(S) = 1 - \prod_{u \in N(v) \cap S} (1 - p_{uv}) \tag{11}$$

where p_{uv} is the propagation probability from u to v.

(2) Intra-community propagation: In this phase, the influence will only propagate within communities and the influence propagation in these communities are independent of each other. It is natural to see that the final influence in a community C is a function of: the nodes in this community, the first and second order seeds. For convenience, we will use S' instead of $N(S)$ to denote the second order seeds in the following part of this paper. We define the final influence of community C as $f(S, S', C)$. Then the total influence can be computed by:

$$f(S) = \sum_{C \in \mathcal{C}} f(S, S', C) \tag{12}$$

Based on the above equation, we may define the community-based influence maximization problem as:

Definition 2. (Community-based influence maximization) Given network $G(V, E)$ with non-overlapping community structure $\mathcal{C}$, the community-based influence maximization problem aims to find a subset S^* of k nodes, such that the objective function defined in Eq. (12) is maximized, i.e.,

$$S^* = \arg_S \max f(S) \tag{13}$$

Let $\mathbb{P}_v(S, S', C)$ denote the probability that node v in community C is finally activated. Then $f(S, S', C)$ can be computed as follows:

$$f(S, S', C) = \sum_{v \in C} \mathbb{P}_v(S, S', C) \tag{14}$$

Although the form of Eq. (14) is unique, the computation of $\mathbb{P}_v(S, S', C)$ is open. It is easy to see that for any $v \in S$, we have $\mathbb{P}_v(S, S', C) = 1$. For any $v \in S'$, i.e., $v \in N(S)$, we have $\mathbb{P}_v(S, S', C) = 1 - \Pi_{u \in N(v) \cap S}(1 - p_{uv})$. The key is the computation of $\mathbb{P}_v(S, S', C)$ for the rest vertices.

4.2. Computation of $\mathbb{P}_v(S, S', C)$

As an approximation approach, the value of $\mathbb{P}_v(S, S', C)$ for any node v not in S and S', i.e., $v \in V \backslash \{S \cup S'\}$, can be computed in arbitrary way, here we introduce three methods:

[1] Although we only consider undirected unweighted graph here, our framework can easily be extended to directed and weighted networks.

4.2.1. Monte–Carlo simulations

Repeat the Monte–Carlo simulation R times within communities. Note that since our model has both first and second order seeds, the MC simulation is a little different from the traditional way. Specifically, in each simulation, at $t = 0$, we activate each node in S and activate each node in S' with probability $\mathbb{P}_v(S)$ as defined in (11). Then the influence starts propagating within this community where cross-community propagation is forbidden. Assume that among the R simulations, vertex v is successfully activated for $R(v)$ times, then $\mathbb{P}_v(S, S', C)$ is approximated by:

$$\mathbb{P}_v(S, S', C) = R(v)/R \tag{15}$$

4.2.2. Constant value

A simple and naive choice is for each node $v \in C\backslash\{S\cup S'\}$, we set a constant value for $\mathbb{P}_v(S, S', C)$, so we have:

$$\mathbb{P}_v(S, S', C) = \begin{cases} 1 & v \in S \\ 1 - \prod_{u\in N(v)\cap S}(1 - p_{uv}) & v \in S' \\ \rho & v \in C \setminus \{S \cup S'\} \end{cases} \tag{16}$$

From Eq. (14), the total influence of community C is:

$$f(S, S', C) = |S \cap C| + \sum_{v\in C\cap S'}(1 - \prod_{u\in N(v)\cap S}(1 - p_{uv})) + \rho|C \setminus \{S \cup S'\}| \tag{17}$$

In this case, the rest contribution of each community is proportional to its rest nodes.

4.2.3. Constant sum

Another choice is for each node $v \in C\backslash\{S\cup S'\}$, we set the sum of $\mathbb{P}_v(S, S', C)$ for those nodes as a constant, i.e., $\sum_{v\in C\setminus\{S\cup S'\}} \mathbb{P}_v(S, S', C) = \rho$, so we have:

$$\mathbb{P}_v(S, S', C) = \begin{cases} 1 & v \in S \\ 1 - \prod_{u\in N(v)\cap S}(1 - p_{uv}) & v \in S' \\ \frac{\rho}{|C\setminus\{S\cup S'\}|} & v \in C \setminus \{S \cup S'\} \end{cases} \tag{18}$$

The total influence of community C is:

$$f(S, S', C) = |S \cap C| + \sum_{v\in C\cap S'}(1 - \prod_{u\in N(v)\cap S}(1 - p_{uv})) + \rho \tag{19}$$

In this case, the rest contribution of each community is a constant.

4.3. Simplification under the weighted cascade model

WC (Weighted Cascade) model is an extension of the classic IC model, the only difference is: in the WC model, an active vertex u activates its inactive neighbor v and succeed with probability $1/k_v$, which is determined by v's degree. Here we show how the objective function $f(S)$ as defined in Eq. (12) can be simplified under this model when we use constant sum to compute the $\mathbb{P}_v(S, S', C)$ value.

Theorem 3. *Under the WC diffusion model, if we use constant sum as defined in Eq. (18) to compute the $\mathbb{P}_v(S, S', C)$ value, then the overall influence can be evaluated by:*

$$g(S) = |N(S)| + \gamma|NC(S)| \tag{20}$$

where γ is a parameter.

Proof. It is easy to see that the number of connections emit from S is $\Sigma_{v\in S}k_v$, and these edges point to $|N(S)|$ vertices, i.e., to the second order seeds. For any vertex $v \in N(S)$, if we randomly select an edge from those emit from S, then the probability that this edge points to v is proportional to v's degree [43], let us denote this probability as $p = \alpha \cdot k_v$. Obviously, the event that an edge from S points to v is a Bernoulli experiment and thus the expected number of connections from S to v is $p\sum_{u\in S}k_u = \alpha k_v \sum_{u\in S}k_u$. Then for any $v \in N(S)$, from Eq. (11), $\mathbb{P}_v(S)$ can be approximated by:

$$\mathbb{P}_v(S) = 1 - \left(1 - \frac{1}{k_v}\right)^{\alpha k_v \sum_{u\in S}k_u} \simeq \alpha \sum_{u\in S} k_u \tag{21}$$

The second equation comes from the fact that if $x \ll 1$, then we have $1 - (1 - x)^n \simeq nx$. From this formula, we see that once the seed set is selected, then $\mathbb{P}_v(S)$ for each $v \in N(S)$ can be viewed as a constant.

If we use constant sum to compute the $\mathbb{P}_v(S, S', C)$ value, then from Eq. (18), the total influence of community C can be approximated by:

$$f(S, S', C) = |S \cap C| + \alpha \sum_{u\in S} k_u|N(S) \cap C| + \rho \tag{22}$$

From Eq. 12, the total influence can be approximated by:

$$\begin{aligned} f(S) &= \sum_{C\in\mathcal{C}} f(S, S', C) \\ &= |S| + \alpha \sum_{u\in S} k_u|N(S)| + \rho|NC(S)| \\ &\simeq \alpha \sum_{u\in S} k_u|N(S)| + \rho|NC(S)| \\ &= \left(\alpha \sum_{u\in S} k_u\right)\left(|N(S)| + \frac{\rho}{\alpha \sum_{u\in S} k_u}|NC(S)|\right) \end{aligned} \tag{23}$$

The third equation comes from the fact that $|S| \ll |N(S)|$. Let $\gamma = \rho/(\alpha \sum_{u\in S} k_u)$ and eliminate the coefficient $\alpha\Sigma_{u\in S}k_u$ of $|N(S)|$, we finally get Eq. (20), where $f(S) = (\alpha \sum_{u\in S} k_u)g(S)$ □

We should note that although we use $g(S)$ to evaluate the total influence, $g(S)$ is not an approximation of $f(S)$ as we have eliminated the coefficient $\alpha\Sigma_{u\in S}k_u$ of $f(S)$ in Eq. (23).

Eq. (20) has two advantages: (i) it can be easily and efficiently computed; and (ii) it is submodular. Now we give the proof of submodularity as follows.

Theorem 4. *The objective function $g(S)$ defined in Eq. (20) is submodular.*

Proof. To prove $g(S)$ is submodular, we only have to prove that $g_1(S) = |N(S)|$ and $g_2(S) = |NC(S)|$ are both submodular. Firstly, we prove that $g_1(S)$ is submodular, i.e., the following inequation satisfies:

$$g_1(S \cup v) - g_1(S) \geq g_1(T \cup v) - g_1(T), \forall v \in V, S \subseteq T \tag{24}$$

We prove it by considering all cases:

If $v \in S$, we have $g_1(S \cup v) - g_1(S) = g_1(T \cup v) - g_1(T) = 0$, in Eq. (24) satisfies.

If $v \in T\backslash S$, we have $g_1(S \cup v) - g_1(S) \geq 0 = g_1(T \cup v) - g_1(T)$, in Eq. (24) satisfies.

If $v\notin T$, then $v\notin S$. We have $g_1(S \cup v) - g_1(S) = |N(S \cup v) \setminus N(S)| = |N(v) \setminus N(S)|$. Similarly, we have $g_1(T \cup v) - g_1(T) = |N(v) \setminus N(T)|$. Since $S\subseteq T$, we have $N(S)\subseteq N(T)$, so we have $N(v)\backslash N(S)\supseteq N(v)\backslash N(T)$, i.e., $g_1(S \cup v) - g_1(S) \geq g_1(T \cup v) - g_1(T)$, in Eq. (24) satisfies.

To sum up, $g_1(S) = |N(S)|$ is submodular. Similarly, we can prove that $g_2(S) = |NC(S)|$ is submodular. So $g(S) = g_1(S) + \gamma \cdot g_2(S)$ is submodular. □

4.4. Seed nodes selection

Based on the theory of submodularity [6], if a function is submodular, then the "hill-climbing" greedy algorithm provides a factor of $(1 - 1/e)$ approximiation to the optimal solution. Since our

object function $g(S)$ is submodular, we apply the greedy algorithm to select the seed nodes. The main advantage of our algorithm over Kempe et al.'s greedy algorithm is that our objective function $g(S)$ can be easily and efficiently computed, while Kempe et al.'s algorithm requires tens of thousands of Monte-Carlo simulations to compute the object value. The pseudocode of our algorithm is shown in Algorithm 2. Initially the seed set S is empty. Each time

Algorithm 2 The CoFIM algorithm.

Require:
Network G, community structure $\mathcal{C}$, number of seed nodes k, parameter γ
Ensure:
Seed set S
1: Initialize: Let $S \leftarrow \Phi$.
2: **for** $i = 1$ to k **do**
3: $v = \arg_u \max\{g(S \cup \{u\}) - g(S)\}$
4: // $g(*)$ is computed using Eq. 20
5: $S \leftarrow S \cup \{v\}$
6: **end for**
7: **return** S

we add a node v that brings the maximum marginal gain of objective function $g(S)$ to S, until the size of S reaches k. Similar to the traditional greedy algorithm, we apply the CELF strategy [7] to accelerate our algorithm.

4.5. Time complexity

Here we give the time complexity analysis of our CoFIM algorithm.

Theorem 5. *The worst-case time complexity of Algorithm 2 is* $O(k^2 \cdot n \cdot k_{max})$, *where* k *is the number of seed nodes,* $n = |V|$ *is the number of network nodes, and* k_{max} *is the maximum node degree.*

Proof. We see that the CoFIM algorithm iteratively selects the new seed node with the maximum marginal gain of $g(S)$ and add it to the seed set S, which is empty at step $t = 0$. In each iteration, under the worst case, we have to traverse all the candidates $v \in V \backslash S$ to find the new seed. Given arbitrary seed set T, the time complexity of computing $g(T)$ is $N(T)$. Since our algorithm tends to select nodes with high degree values and $|T| \leq k$ is satisfied for all round of iterations, we have $N(T) \leq \sum_{v \in T} N(v) \leq \sum_{v \in T} k_{max} = |T| k_{max} \leq k \cdot k_{max}$. So the worst-case time complexity of one iteration is $O(|V \setminus S| \cdot k \cdot k_{max}) = O(n \cdot k \cdot k_{max})$. Since the CoFIM algorithm iteratively selects k seed nodes, the overall worst-case time complexity of the CoFIM algorithm is $O(k^2 \cdot n \cdot k_{max})$. □

5. Evaluation

5.1. Dataset

5.1.1. Real-world networks

We first evaluate the performance of our community-based influence maximization algorithm on nine real world datasets, which provide a spectral of application areas and ranges from medium size to mega-scale, reflecting the volume and variety properties of big data. The largest dataset contains 3.1 million nodes and about 117 million edges. Two medium-sized datasets (NetHEPT and NetPHY) are downloaded from the website[2] provided by Chen et al. [11], while the other seven larger datasets are downloaded from the SNAP website[3] maintained by Jure Leskovec.

NetHEPT: The NetHEPT dataset is provided by the e-print platform arXiv (www.arxiv.org/). The network contains the collaboration relationships among authors in the area High Energy Physics Theory. If an author i co-authors at least one paper with author j, then an undirected edge will be created between i and j. The dataset contains 15K nodes and 31K edges.

NetPHY: The NetPHY dataset also comes from the arXiv platform, including the collaboration relationships among authors in the area of Physics. The network includes 37K nodes and 174K edges.

Epinions: The Epinions dataset [44] contains the who-trust-whom data from a consumer review site Epinions.com. In this website customers can submit online reviews on the products and other users may choose whether or not to trust the review. Users form the nodes while their trust relationships form the edges. We simply preprocessed the data by removing the repeated edges, resulting in a network of 76K nodes and 406K edges.

Amazon: The Amazon dataset is obtained from SNAP (Stanford Large Network Dataset Collection), an online libaray provided by Leskovec et al. [45]. They crawled the data from the online shopping website Amazon (www.amazon.com/). If a product i is frequently co-purchased with product j, then an undirected edge is created between i and j. The network contains 335K nodes and 926K edges.

DBLP: The DBLP online library is a large collection of papers in computer science. The dataset provides a co-authorship network among research workers. If two authors have collaborated on at least one paper, an undirected edge is created between them. The network consists of 317K nodes and 1M edges.

Patent: The Patent dataset [46] includes all the utility patents granted during a period of 37 years (from January 1, 1963 to December 30, 1999). Each node represents a patent while each edge represents a citation relationship between two patents, resulting into a graph of about 3.8M nodes and 16.5M edges.

Pokec: Pokec represents the popular on-line social network in Slovakia. It has been provided for more than 10 years and connects more than 1.6 million people. The dataset contains the whole social network among Slovakia users, including about 1.6M nodes and 22.3M connections, provided by Takac et al. [47].

LiveJournal: LiveJournal is a free on-line community with almost 10 million members, who are allowed to maintain their journals, indivdual and group blogs. Users may also declare friendships with other members. The graph consists of about 4M members with 35M pairs of friendship relationships, provided by Backstrom et al. [48].

Orkut: Orkut is a free on-line social network where users form friendship with each other. It allows users form a group which other members can then join. The graph contains about 3.1M nodes as users and 117M edges as their friendship connections, provided by Mislove et al. [49].

We treat all the edges as undirected ones. Table 1 summaries the statistical properties of these datasets.

5.1.2. Synthetic networks

We also evaluate the performance of our CoFIM algorithm through synthetic networks. Since our framework is based on community structure, we choose to use the LFR algorithm [30] to generate synthetic benchmark networks, due to its following advantages:

(1) Community networks: The algorithm is able to generate networks with configurable ground truth community structure.

[2] http://research.microsoft.com/enus/people/weic/graphdata.zip.

[3] http://snap.stanford.edu/data/.

Table 1
Summary of nine real world datasets.

Dataset	NetHEPT	NetPHY	Epinions	Amazon	DBLP	Patent	Pokec	LiveJournal	Orkut
# Ndoes	15K	37K	76K	335K	317K	3.8M	1.6M	4M	3.1M
# Edges	31K	174K	406K	926K	1M	16.5M	22.3M	35M	117M
Max. Degree	64	178	3,044	290	343	793	14,854	14,724	33,313
Avg. Degree	4.12	9.38	10.69	4.34	6.62	8.74	27.22	17.01	76.17
# Communities	2,262	8034	10,307	12,326	11,933	15,563	2520	116,288	5118
Max. Com. Size	661	88	11,220	1762	13,913	133,687	209,159	299,915	764,325
Min. Com. Size	2	1	1	1	1	1	1	1	1
Avg. Com. Size	6.73	4.62	7.36	27.17	26.57	242.55	647.94	34.48	600.32
Modularity	0.836	0.808	0.727	0.84	0.763	0.787	0.771	0.724	0.856
Parameter γ	3	4	3	3	3	3	6	3	3

(2) Power-law degree distribution: The algorithm generates networks whose degree follows a power-law distribution. The community size of the network also follows power-law distribution. The power-law distribution (i.e., the scale-free property) is a common property shared by most real-world networks, making our results more practicable.
(3) Highly configurable network parameters: The algorithm allows tweaking a spectrum of parameters, such as the number of nodes N, the average degree $\langle k \rangle$, the maximum degree k_{max}, the power exponent of the degree distribution τ_1 and community size distribution τ_2, the minimum/maximum size of the communities s_{min}/s_{max}, the mixing parameter μ (which is the average percentage of neighbors that does not belong to the same community for each node). These parameters provide us the flexibility to generate networks with various community structures.

By experimenting on the LFR synthetic networks, we can see how different network community structures affect the performance of our CoFIM algorithm.

5.2. Baseline algorithms

We compare the performance of our CoFIM algorithm with five baseline algorithms, which include three state-of-the-art algorithms having been successfully applied on large-scale networks and two node centrality based heuristic algorithms. We do not include the traditional greedy algorithm due to its extremely low time efficiency in handling large-scale networks. We also do not consider the random selection algorithm as its results make little sense.

- **IPA**: The IPA (Independent Path Algorithm) is an influence path based algorithm proposed by Kim et al. [10]. The algorithm assumes that influence only propagate through paths whose influence probability is larger than a given threshold and different influence paths are independent of each other. Based on this assumption, the influence spread can be efficiently computed in a parallel manner. The algorithm requires a parameter θ to control the length of influence paths. In our experiments we set $\theta = 1/320$, as recommended by the authors in their paper [10].
- **TIM+**: The TIM+ (Two-phase Influence Maximization) algorithm proposed by Tang et al. [50] is an influence maximization method that utilizes the idea of Borgs et al.'s *RIS* algorithm [51]. The algorithm contains two phases: (i) parameter estimation and (ii) node selection. The first phase computes a lower-bound of the maximum expected spread among all size-k node set and derives a parameter θ, while the second phase samples θ random RR [51] sets from G to derives the best node set S^*. The algorithm has been successfully applied on mega-scale networks to show its scalability. The algorithm has a parameter ε which controls its accuracy. In our experimental study we set $\varepsilon = 0.1$ as recommended.
- **IMM**: The IMM (Influence Maximization via Martingales) algorithm proposed by Tang et al. [52] is an extension of their previously proposed TIM+ algorithm. By taking advantage of martingales (a classic statistical tool), IMM algorithm is able to provide the same worst-case guarantees as TIM+, while significantly reduce its computational costs, which is done by adding an intermediate step between TIM+'s two phase approach that heuristically refines parameter θ into a tighter lower bound. Similar to TIM+, we set the parameter $\varepsilon = 0.1$ for the IMM algorithm.
- **SD**: The SD (Single Discount) algorithm is a node centrality-based algorithm proposed by Chen et al. [11]. In each iteration the algorithm selects the node with the highest degree and add it to the seed set. However, once a node is selected, the degree of each of its neighbors will be reduced (discounted) by 1.
- **Degree**: The simple degree based algorithm selects top k nodes with the highest degree.

5.3. Evaluation metrics

Similar to other state-of-the-art algorithms [9,10], we employ two metrics to evaluate the performance of our algorithm.

- **Infuence spread**: Given seed set S, influence spread is defined as the number of expected active nodes after the diffusion process stopped. It is used to evaluate the accuracy of an influence maximization algorithm. Higher influence spread value indicates more accurate algorithm. Since the computation of exact influence spread is #P hard [9], a general way to obtain the influence spread value is by running Monte-Carlo simulations. Similar to [5], we repeat 10,000 MC simulations to compute the influence spread for any given seed set S.
- **Running time & Memory usage**: Running time is defined as the time for selecting k seed nodes (not include the time of building graphs) while memory usage is defined as the overall bytes of memory used by different algorithms in finding the seed set S. They are widely used to evaluate the time and space efficiencies of an algorithm. In real-world big data applications which always relate to mega-scale networks, time and space efficiencies are important issues that researchers concern about.

5.4. Experimental procedure

Community detection: Since our community-based algorithm relies on the community structure of a network, to obtain the community structure on real-world networks, we apply the Louvain [53] algorithm which has been proved to have good performance in community detection [23]. The Louvain algorithm is based on *modularity* optimization, where *modularity* is a metric to evaluate the quality of a community structure. It is propsed by Newman et al. [54], and usually denoted by Q, which varies between -1

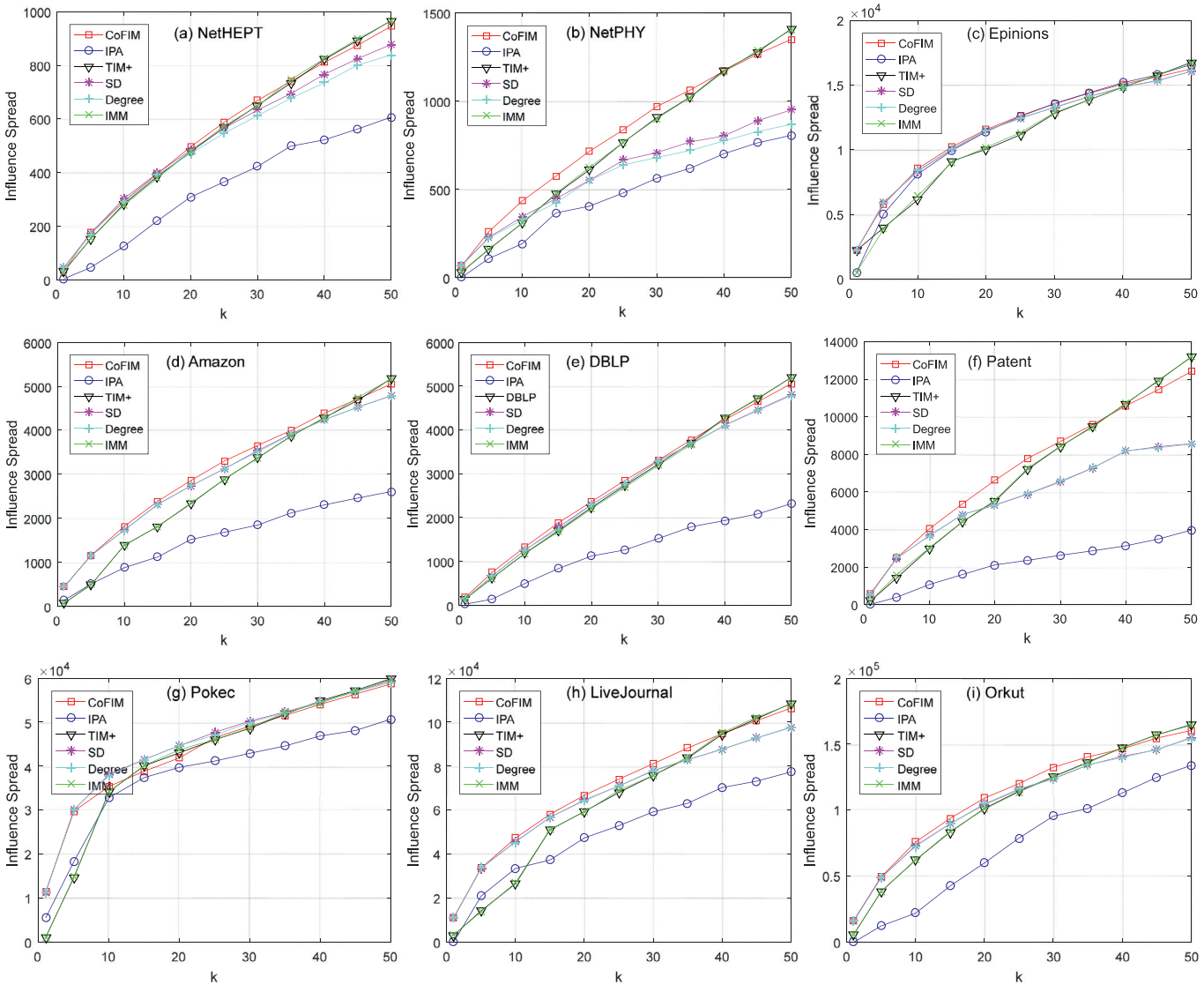

Fig. 1. Influence spread on nine real-world networks

and 1. Intuitively, the more connections within communities and less connections among communities, the higher Q is. Table 1 summaries the community information obtained by running Louvain algorithm on the nine real world datasets. From Eq. (20) we see that our algorithm requires a parameter γ, which is also shown in this table. The parameter values are decided through small scale experiments.

Diffusion model: From Section 4.3 we know that the diffusion model we use in this paper is the WC (weighted cascade) model, i.e., the propagation probability from u to v is $p_{uv} = 1/k_v$, which only relies on the degree of v. The influence spread value is computed by repeating 10,000 Monte-Carlo simulations.

Experimental environment: The experiments are carried out on a computer with 3.5GHz Intel Xeon E3-1246 v3 CPU and 32GB memory. Our codes[4] are programmed in C/C++ and the programs are in single process and single thread. The source codes of all the baseline algorithms provided by their authors are also written in C/C++, making the results more equitable.

6. Results

6.1. Results on real-world networks

6.1.1. Influence spread

We first compare the influence spread of different algorithms on nine real world datasets, as shown in Fig. 1, where x-axis represents the number of seed nodes we want to find while y-axis represents the overall influence spread. From the results on the nine real-world datasets, we see that our CoFIM algorithm is always among the ones (together with TIM+ and IMM) providing the best performance in terms of influence spread. The IPA algorithm shows the worst performance on all the networks except for the Epinions dataset, as shown in Fig. 1(c). The node centrality based algorithms (SD and Degree), though performs well on some datasets, cannot provide any performance guarantee. For example, on three datasets: NetHEPY, NetPHY, and Patent, their influence spread val-

[4] The source code can be downloaded from: https://sourceforge.net/projects/cofim-com-based-fast-influ-max/.

Table 2
Running time of different algorithms on nine real world datasets ($k = 50$).

Run Time(s)	NetHEPT	NetPHY	Epinions	Amazon	DBLP	Patent	Pokec	LiveJournal	Orkut
CoFIM	**≤ 1**	**≤ 1**	**≤ 1**	**≤ 1**	**≤ 1**	3	9	12	74
IPA	3	2	20	5	69	147	430	1154	605
TIM+	4	12	27	73	79	939	494	485	1939
IMM	**≤ 1**	3	4	15	15	176	84	83	290
SD	**≤ 1**	**≤ 1**	**≤ 1**	**≤ 1**	**≤ 1**	**≤ 1**	**≤ 1**	**≤ 1**	**≤ 1**
Degree	**≤ 1**	**≤ 1**	**≤ 1**	**≤ 1**	**≤ 1**	**≤ 1**	**≤ 1**	**≤ 1**	**≤ 1**

Table 3
Memory usage of different algorithms on nine real world datasets ($k = 50$).

Memory	NetHEPT	NetPHY	Epinions	Amazon	DBLP	Patent	Pokec	LiveJournal	Orkut
CoFIM	**7M**	**22M**	**47M**	**116M**	**126M**	1.8G	2.2G	3.5G	11G
IPA	263M	284M	1.2G	257M	3.7G	**1.7G**	14.2G	31.3G	22.5G
TIM+	450M	861M	318M	3.1G	3.0G	24.7G	3.9G	6.2G	11G
IMM	95M	187M	84M	712M	642M	5.3G	**1.9G**	**2.4G**	**5.5G**
SD	**7M**	**22M**	**47M**	**116M**	**126M**	1.8G	2.2G	3.5G	11G
Degree	**7M**	**22M**	**47M**	**116M**	**126M**	1.8G	2.2G	3.5G	11G

ues, as shown in Fig. 1(a),(b),(f), are significantly inferior to that of CoFIM, TIM+ and IMM algorithms. The difference between node centrality based algorithms and the CoFIM algorithm indicates that taking advantage of community information can significantly improve the accuracy of influence maximization algorithms.

Since the IMM algorithm is an extension of the TIM+ algorithm providing the same influence guarantee, they exhibit the same influence spread value. Although the influence spread values of TIM+ and IMM algorithms are slightly above that of our CoFIM algorithm's with $k = 50$, our algorithm shows its robustness across different values of k. When k is small (e.g., $k \leq 20$), on several datasets (NetPHY, Epinions, Amazon, Patent, and LiveJournal) we see that the influence spread values of TIM+ and IMM algorithms are much lower than our CoFIM algorithm. For example, on LiveJournal dataset with $k = 10$, CoFIM algorithm achieves a influence spread value of 47,322, while the values of TIM+ and IMM algorithms are 26,598 and 27,024 respectively, only about half of ours as shown in Fig. 1(h).

Overall, from the results on the nine real-world networks, our CoFIM algorithm shows its effectiveness and robustness in finding top influential seed nodes as compared with the state-of-the-art algorithms.

6.1.2. Running time and memory usage

The time efficiency is a key issue that many researchers concern about in big data applications. Table 2 shows the running time of different algorithms on the nine real-world datasets. Here the running time is the time of selecting $k = 50$ seed nodes. From the results we see that the CoFIM algorithm is very competitive in its time efficiency, requiring less than one second on most datasets, including the DBLP dataset with about one million edges. The node centrality based heuristic algorithms, though run fast, cannot provide any performance guarantee in terms of influence spread, as previously shown in Fig. 1. On the Orkut datset, the largest one with 3.1M nodes and 117M edges, our CoFIM algorithm requires only 74 s (slightly over one minute) to find the seed nodes, about four times faster than the IMM algorithm and more than 26 times faster than the TIM+ algorithm. On the Patent dataset, the CoFIM algorithm is more than 300 times faster than the TIM+ algorithm, which shows its extremely high time efficiency.

The memory usage is another issue concerned by researchers in big data applications. Table 3 shows the memory usage of different algorithms on the nine real-world datasets. Again our CoFIM algorithm exhibits its high scalability in dealing with mega-scale networks. From all of the datasets we observe that the CoFIM algorithm requires no extra memory as compared with the simple node centrality based algorithms (SD and Degree). On many of the datasets, the CoFIM algorithm requires the lest amount of memory as compared with other state-of-the-art algorithms. For example, on the NetHEPT dataset, the memory usage of CoFIM algorithm in 7M, only about 7% and 1.6% to that of the IMM and TIM+ algorithms respectively. On very large-scale networks, the memory usage of our CoFIM algorithm becomes larger than the IMM algorithm. We think this is mainly due to the data structure we use to represent the graphs.

6.1.3. Impact of algorithm parameter γ

As shown in Eq. (20), our CoFIM algorithms depends on a parameter γ, which weights the importance of community information in selecting seed nodes. Larger γ indicates that greater importance is put on community information. We evaluate how the algorithm parameter γ affects the influence spread on the real-world datasets, as shown in Fig. 2. The x-axis represents the parameter value γ while the y-axis represents the influence spread in selecting $k = 50$ seed nodes. Here we only give the results on three datasets (NetPHY, DBLP, and LiveJournal) since the results on other datasets are similar. Fig. 2 shows that by selecting an appropriate value of parameter γ, the overall influence spread can be maximized. The parameter values of γ corresponding to the maximum influence spread on the three datasets are $\gamma = 6$ (NetPHY), $\gamma = 3$ (DBLP), and $\gamma = 3$ (LiveJournal) respectively. However, if the parameter γ is too large (e.g., $\gamma > 10$), the influence spread value can be severely affected. So in practice, we should be careful in selecting the value of γ.

6.2. Results on synthetic networks

The results on real-world datasets have shown the high scalability of our CoFIM algorithm. In order to take a deep insight into the algorithm, we also experiment on synthetic networks, which are generated using the LFR [30] algorithm.

6.2.1. The impact of community structure

As our framework is based on community structure, we want to see how the network community structure affects the performance of the CoFIM algorithm. To do this, we choose the LFR algorithm to generate synthetic benchmark networks with configurable ground truth community structure. As mentioned in Section 5.1.2, the LFR algorithm accepts the following parameters: the number of nodes N, the average degree $\langle k \rangle$, the maximum degree k_{max}, the power exponent of the degree distribution τ_1 and community size

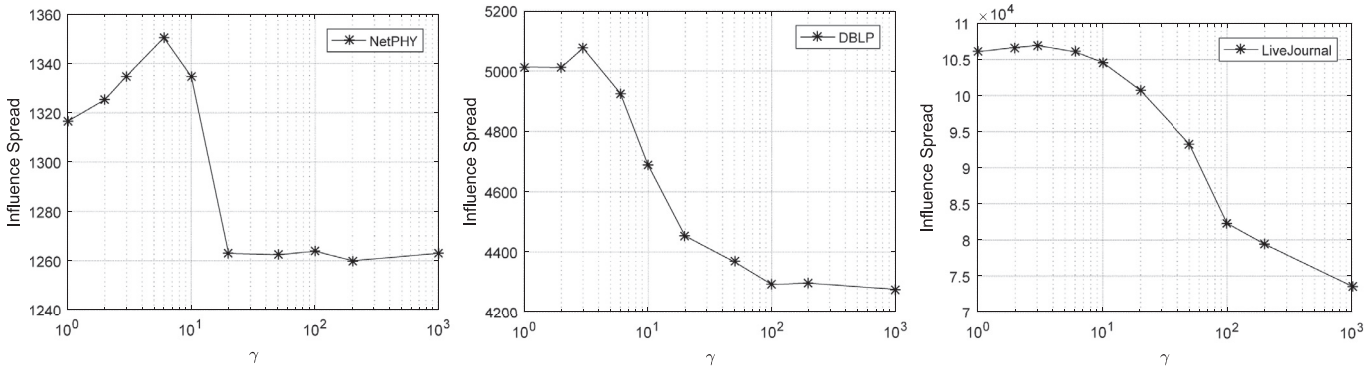

Fig. 2. The impact of parameter γ on influence spread on three real-world networks (k=50).

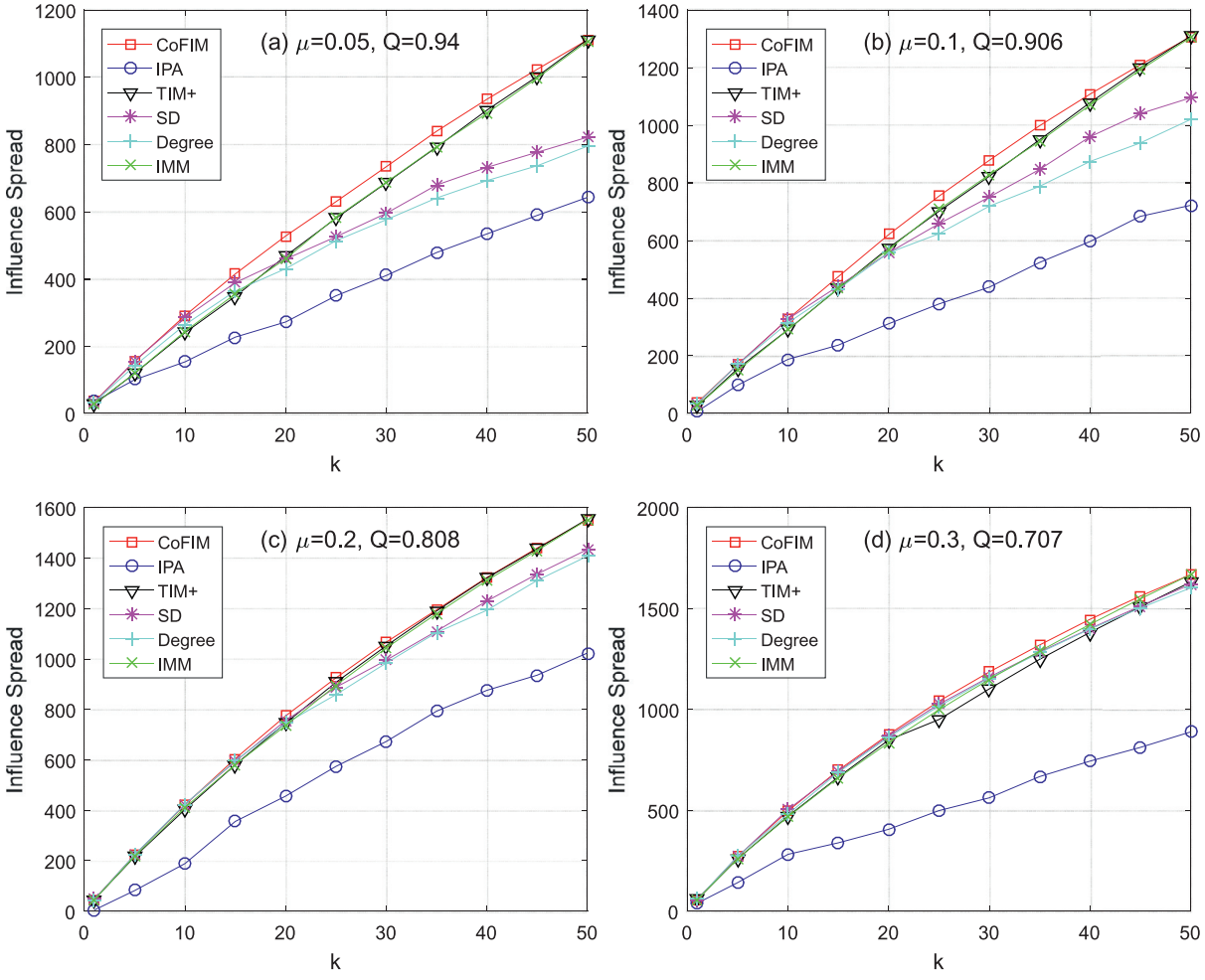

Fig. 3. Influence spread on four LFR synthetic networks with different community structures. The network parameters are: $N = 10,000$, $\langle k \rangle = 15$, $k_{max} = 100$, $\tau_1 = 2$, $\tau_2 = 1$, $s_{min} = 20$, $s_{max} = 100$. The community structure is regulated by parameter μ: (a) $\mu = 0.05$; (b) $\mu = 0.1$; (c) $\mu = 0.2$; (d) $\mu = 0.3$.

distribution τ_2, the minimum/maximum size of the communities s_{min}/s_{max}, the mixing parameter μ. Among them four parameters (τ_2, s_{min}, s_{max}, μ) can be used to regulate the community structure of the generated networks. In our experiments we only consider the parameter μ since it directly determines the community style, smaller μ means more connections within communities than across communities, i.e., stronger community structure. Fig. 3 gives the results on four synthetic LFR networks with different community styles: (a) extremely strong community ($\mu = 0.05$); (b) strong community ($\mu = 0.1$); (c) medium-strong community ($\mu = 0.2$); and (d) weak community ($\mu = 0.3$). The other network parameters are kept unchanged as: $N = 10,000$, $\langle k \rangle = 15$, $k_{max} = 100$, $\tau_1 = 2$, $\tau_2 = 1$, $s_{min} = 20$, $s_{max} = 100$. We do not generate very large synthetic networks due to the low time efficiency of the LFR algorithm in generating such networks. We also give the results of other state-of-the-art algorithms as comparisons.

From the results on the four networks our CoFIM algorithm again shows its advantage in finding influential seed nodes over other algorithms as it always achieves the best performance in terms of influence spread. Moreover, it is observed that the com-

Table 4
Running time and memory usage of different algorithms on three LFR synthetic networks ($k = 50$).

Time (s) & Memory	LFR ($\mu = 0.05$)	LFR ($\mu = 0.1$)	LFR ($\mu = 0.2$)	LFR ($\mu = 0.3$)
CoFIM	**< 1 / 11M**	**< 1 / 11M**	**< 1 / 11M**	**< 1 / 11M**
IPA	1 / 109M	1 / 112M	1 / 128M	2 / 150M
TIM+	5 / 284M	5 / 270M	5 / 274M	6 / 271M
IMM	1 / 71M	1 / 63M	1 / 61M	1 / 65M
SD	**< 1 / 11M**	**< 1 / 11M**	**< 1 / 11M**	**< 1 / 11M**
Degree	**< 1 / 11M**	**< 1 / 11M**	**< 1 / 11M**	**< 1 / 11M**

munity style can significantly affect the effectiveness of the CoFIM algorithm. In networks with very strong community structure (as shown in Fig. 3(a)), by taking advantage of community information, the CoFIM algorithm significantly outperforms other state-of-the-art algorithms. In contrast, on networks with weak community structure (as shown in Fig. 3(d)), the difference between the CoFIM and other algorithms becomes inconspicuous. Holding the fact that our framework relies the hypothesis of strong community structure, the results are intuitive.

6.2.2. Running time and memory usage

The running time and memory usage of different algorithms on the four synthetic networks are given in Table 4. On all the four synthetic networks, the CoFIM algorithm (together with the SD and Degree heuristic algorithms) requires the least time and memory in finding $k = 50$ seed nodes. The memory usage of the CoFIM algorithm is the same as that of the SD and Degree algorithms, while it is about 6, 10, and 25 times more efficient than the IMM, IPA and TIM+ algorithms, respectively. The results are consistent with our findings on real-world datasets.

In sum, by experimenting on both synthetic and real-world networks, our CoFIM algorithm shows great advantage over state-of-the-art algorithms in both efficiency and effectiveness.

7. Conclusion

Influence maximization is a classic propagation optimization problem studied in the area of social network analysis and viral marketing. The simple greedy algorithm, proposed by Kempe et al., though provides a factor of $(1 - 1/e - \varepsilon)$ approximation to the optimal solution, cannot be applied to large-scale networks due to its low time efficiency. Other submodularity-based or node centrality-based heuristics algorithms, either achieve limited improvement in time efficiency, or cannot provide any performance guarantee in terms of accuracy. To solve the problem of traditional influence maximization algorithms, in this paper we propose community-based influence maximization framework, which divides the propagation process into two phases: (1) seeds expansion; and (2) intra-community propagation. Based on our framework, we derive a simplified evaluation of the influence spread under the weighted cascade model and theoretically prove that our objective function is submodular. Then we further proposed an efficient greedy algorithm to select the top k seed nodes. We evaluate the performance of our community-based algorithm on nine real world datasets and four synthetic networks with ground truth community structure. Experimental results indicate that our algorithm significantly outperforms state-of-the-art algorithms in both efficiency and effectiveness.

To the best of our knowledge, our approach makes the first effort to combine community-structure with influence diffusion models. We believe our work will provide deep insight into the studies of influence maximization problem in the future.

Our work can be extended in several directions in the future. First, the effectiveness of our community-based framework, though experimentally evaluated, requires more theoretical validation. In the future, we will study how the community structure, e.g., the modularity value, affects the effectiveness of our framework. Second, our framework is based on non-overlapping community structure, while real world communities may overlap [17]. We should study how to make our framework support overlapping community structure. Last but not least, the simplified objective function in our framework is derived based on the weighted cascade model. In the future we should study similar simplifications under other diffusion models, such as the linear threshold model.

Acknowledgements

We appreciate the anonymous reviewer's valuable comments. This work was partly supported by the Fundamental Research Funds for the Central Universities (No. 0903005203400, 0216005202068). This work was also partially supported by the foundation from the "China Equipment and Resource Sharing" project (No. 025-226009002, 226009003, Tsinghua University). This work was also partially supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (CityU 149412), and City University of Hong Kong SRG 7004287 and 7004142. All opinions are those of the authors and do not necessarily reflect the views of the funding agencies.

References

[1] J. Goldenberg, B. Libai, E. Muller, Talk of the network: a complex systems look at the underlying process of word-of-mouth, Mark. Lett. 12 (3) (2001) 211–223.
[2] J. Leskovec, L.A. Adamic, B.A. Huberman, The dynamics of viral marketing, ACM Trans. Web (TWEB) 1 (1) (2007) 5.
[3] J.J. Brown, P.H. Reingen, Social ties and word-of-mouth referral behavior, J. Consumer Res. (1987) 350–362.
[4] V. Mahajan, E. Muller, F.M. Bass, New product diffusion models in marketing: a review and directions for research, J. Market. (1990) 1–26.
[5] D. Kempe, J. Kleinberg, É. Tardos, Maximizing the spread of influence through a social network, in: ACM 9th SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2003, pp. 137–146.
[6] G.L. Nemhauser, L.A. Wolsey, M.L. Fisher, An analysis of approximations for maximizing submodular set functionsi, Math. Program. 14 (1) (1978) 265–294.
[7] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, N. Glance, Cost-effective outbreak detection in networks, in: ACM 13th SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2007, pp. 420–429.
[8] M. Kimura, K. Saito, Tractable models for information diffusion in social networks, in: Knowledge Discovery in Databases (PKDD), Springer, 2006, pp. 259–271.
[9] W. Chen, C. Wang, Y. Wang, Scalable influence maximization for prevalent viral marketing in large-scale social networks, in: ACM 16th SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2010, pp. 1029–1038.
[10] J. Kim, S.-K. Kim, H. Yu, Scalable and parallelizable processing of influence maximization for large-scale social networks, in: IEEE 29th International Conference on Data Engineering (ICDE), IEEE, 2013, pp. 266–277.
[11] W. Chen, Y. Wang, S. Yang, Efficient influence maximization in social networks, in: ACM 15th SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2009, pp. 199–208.
[12] Y. Wang, X. Feng, A potential-based node selection strategy for influence maximization in a social network, in: Advanced Data Mining and Applications, Springer, 2009, pp. 350–361.
[13] S. Kundu, C. Murthy, S.K. Pal, A new centrality measure for influence maximization in social networks, in: Pattern Recognition and Machine Intelligence, Springer, 2011, pp. 242–247.
[14] M. Sviridenko, A note on maximizing a submodular set function subject to a knapsack constraint, Oper. Res. Lett. 32 (1) (2004) 41–43.

[15] A. Goyal, W. Lu, L.V. Lakshmanan, Celf++: optimizing the greedy algorithm for influence maximization in social networks, in: ACM 20th International Conference Companion on World Wide Web, ACM, 2011, pp. 47–48.
[16] M. Girvan, M.E. Newman, Community structure in social and biological networks, Proc. Nation. Acad. Sci. 99 (12) (2002) 7821–7826.
[17] G. Palla, I. Derényi, I. Farkas, T. Vicsek, Uncovering the overlapping community structure of complex networks in nature and society, Nature 435 (7043) (2005) 814–818.
[18] Y. Dourisboure, F. Geraci, M. Pellegrini, Extraction and classification of dense communities in the web, in: ACM 16th International Conference on World Wide Web, ACM, 2007, pp. 461–470.
[19] U.N. Raghavan, R. Albert, S. Kumara, Near linear time algorithm to detect community structures in large-scale networks, Phys. Rev. E 76 (3) (2007) 036106.
[20] J. Shang, L. Liu, F. Xie, Z. Chen, J. Miao, X. Fang, C. Wu, A real-time detecting algorithm for tracking community structure of dynamic networks, ACM 6th SNA-KDD Workshop, ACM, 2012.
[21] Z. Bu, C. Zhang, Z. Xia, J. Wang, A fast parallel modularity optimization algorithm (fpmqa) for community detection in online social network, Knowl. Based Syst. 50 (2013) 246–259.
[22] H. Sun, J. Huang, X. Zhang, J. Liu, D. Wang, H. Liu, J. Zou, Q. Song, Incorder: incremental density-based community detection in dynamic networks, Knowl. Based Syst. 72 (2014) 1–12.
[23] J. Shang, L. Liu, X. Li, F. Xie, C. Wu, Targeted revision: a learning-based approach for incremental community detection in dynamic networks, Physica A 443 (2016) 70–85.
[24] Z. Liu, B. Hu, Epidemic spreading in community networks, Europhysics Letters (EPL) 72 (2) (2005) 315.
[25] J. Shang, L. Liu, F. Xie, C. Wu, How overlapping community structure affects epidemic spreading in complex networks, in: IEEE 38th International Computer Software and Applications Conference Workshops (COMPSACW), IEEE, 2014, pp. 240–245.
[26] J. Shang, L. Liu, X. Li, F. Xie, C. Wu, Epidemic spreading on complex networks with overlapping and non-overlapping community structure, Physica A 419 (2015) 171–182.
[27] T. Cao, X. Wu, S. Wang, X. Hu, Oasnet: an optimal allocation approach to influence maximization in modular social networks, in: ACM Symposium on Applied Computing, ACM, 2010, pp. 1088–1094.
[28] A. Clauset, M.E. Newman, C. Moore, Finding community structure in very large networks, Phys. Rev. E 70 (6) (2004) 066111.
[29] X. Zhang, J. Zhu, Q. Wang, H. Zhao, Identifying influential nodes in complex networks with community structure, Knowl. Based Syst. 42 (2013) 74–84.
[30] A. Lancichinetti, S. Fortunato, F. Radicchi, Benchmark graphs for testing community detection algorithms, Phys. Review E 78 (4) (2008) 046110.
[31] Y.-C. Chen, W.-Y. Zhu, W.-C. Peng, W.-C. Lee, S.-Y. Lee, Cim: community-based influence maximization in social networks, ACM Trans. Intell. Syst. Technol. (TIST) 5 (2) (2014) 25.
[32] H. Li, S.S. Bhowmick, A. Sun, J. Cui, Conformity-aware influence maximization in online social networks, VLDB J. 24 (1) (2015) 117–141.
[33] E. Even-Dar, A. Shapira, A note on maximizing the spread of influence in social networks, in: Internet and Network Economics, Springer, 2007, pp. 281–286.
[34] E. Even-Dar, A. Shapira, A note on maximizing the spread of influence in social networks, Inf. Process. Lett. 111 (4) (2011) 184–187.
[35] M. Mathioudakis, F. Bonchi, C. Castillo, A. Gionis, A. Ukkonen, Sparsification of influence networks, in: ACM 17th SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2011, pp. 529–537.
[36] H. Lamba, R. Narayanam, A novel and model independent approach for efficient influence maximization in social networks, in: Web Information Systems Engineering (WISE), Springer, 2013, pp. 73–87.
[37] X. Liu, M. Li, S. Li, S. Peng, X. Liao, X. Lu, Imgpu: Gpu-accelerated influence maximization in large-scale social networks, IEEE Trans. Parallel Distrib. Syst. 25 (1) (2014) 136–145.
[38] W. Chen, A. Collins, R. Cummings, T. Ke, Z. Liu, D. Rincon, X. Sun, Y. Wang, W. Wei, Y. Yuan, Influence maximization in social networks when negative opinions may emerge and propagate., in: SDM, vol. 11, SIAM, 2011, pp. 379–390.
[39] X. He, G. Song, W. Chen, Q. Jiang, Influence blocking maximization in social networks under the competitive linear threshold model., in: SDM, SIAM, 2012, pp. 463–474.
[40] H. Zhuang, Y. Sun, J. Tang, J. Zhang, X. Sun, Influence maximization in dynamic social networks, in: Data Mining (ICDM), 2013 IEEE 13th International Conference on, IEEE, 2013, pp. 1313–1318.
[41] J. Kim, W. Lee, H. Yu, Ct-ic: continuously activated and time-restricted independent cascade model for viral marketing, Knowl. Based Syst. 62 (2014) 57–68.
[42] X. Wu, Z. Liu, How community structure influences epidemic spread in social networks, Physica A 387 (2) (2008) 623–630.
[43] R. Pastor-Satorras, A. Vespignani, Epidemic spreading in scale-free networks, Phys. Rev. Lett. 86 (14) (2001) 3200.
[44] M. Richardson, R. Agrawal, P. Domingos, Trust management for the semantic web, in: International semantic Web conference, Springer, 2003, pp. 351–368.
[45] J. Yang, J. Leskovec, Defining and evaluating network communities based on ground-truth, Knowl. Inf. Syst. 42 (1) (2015) 181–213.
[46] J. Leskovec, J. Kleinberg, C. Faloutsos, Graphs over time: densification laws, shrinking diameters and possible explanations, in: Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, ACM, 2005, pp. 177–187.
[47] L. Takac, M. Zabovsky, Data analysis in public social networks, in: International Scientific Conference and International Workshop Present Day Trends of Innovations, 2012, pp. 1–6.
[48] L. Backstrom, D. Huttenlocher, J. Kleinberg, X. Lan, Group formation in large social networks: membership, growth, and evolution, in: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2006, pp. 44–54.
[49] A. Mislove, M. Marcon, K.P. Gummadi, P. Druschel, B. Bhattacharjee, Measurement and analysis of online social networks, in: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement, ACM, 2007, pp. 29–42.
[50] Y. Tang, X. Xiao, Y. Shi, Influence maximization: Near-optimal time complexity meets practical efficiency, in: Proceedings of the 2014 ACM SIGMOD international conference on Management of data, ACM, 2014, pp. 75–86.
[51] C. Borgs, M. Brautbar, J. Chayes, B. Lucier, Maximizing social influence in nearly optimal time, in: Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, 2014, pp. 946–957.
[52] Y. Tang, Y. Shi, X. Xiao, Influence maximization in near-linear time: a martingale approach, in: Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, ACM, 2015, pp. 1539–1554.
[53] V.D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech. 2008 (10) (2008) P10008.
[54] M.E. Newman, M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69 (2) (2004) 026113.