Activity Maximization by Effective Information Diffusion in Social Networks

Zhefeng Wang, Yu Yang, Jian Pei, *Fellow, IEEE*, Lingyang Chu, and Enhong Chen, *Senior Member, IEEE*

Abstract—In a social network, even about the same information the excitements between different users are different. If we want to spread a piece of new information and maximize the expected total amount of excitements, which seed users should we choose? This problem indeed is substantially different from the renowned influence maximization problem and cannot be tackled using the existing approaches. In this paper, motivated by the demand in a few interesting applications, we model the novel problem of activity maximization, and tackle the problem systematically. We first analyze the complexity and the approximability of the problem. We develop an upper bound and a lower bound that are submodular so that the Sandwich framework can be applied. We then devise a polling-based randomized algorithm that guarantees a data dependent approximation factor. Our experiments on four real data sets clearly verify the effectiveness and scalability of our method, as well as the advantage of our method against the other heuristic methods.

Index Terms—Social network, Social Influence, Information diffusion, Activity Maximization

1 Introduction

Consider how one can stimulate the discussion about a topic in a social network as much as possible within a budget. Based on messages between users in an instant messaging network, such as Whatsapp and WeChat, one can model topics and strengths/frequencies of interaction activities between users. In some situations, one may want to raise the awareness of a controversial social issue, such as Trump's pulling the US out of Trans-Pacific Partnership (TPP). Within a budget, one wants to spread the information in the network so that people in the network discuss the issue as much as possible. Which users should we choose to start spreading the words?

We model this problem as *activity maximization*. Given a propagation network, which records user interaction activity strength along each edge, we aim to find an optimal set of seed users under a given budget, such that starting information propagation from the seed users leads to the maximum sum of activity strengths among the influenced users.

Isn't this an instance of the well known and well studied influence maximization problem [1]? The answer is "no" indeed. Influence maximization selects a seed set of nodes within a given budget constraint such that the *expected number of nodes* influenced by information diffusion is maximized. However, to satisfy the requirement that "people in the network discuss the issue as much as possible", we not only want to influence many users, but more importantly also want to maximize the expectation of the sum of strengths of the interaction activities between influenced users. Since the activity strength between users differs from user to user, more influenced users do not necessarily lead to more interaction activities. Figure 1 shows an example. In the figure, the orange nodes and the blue nodes are activated by seed nodes **A** and **B**, respectively. The thick edges carry an activity strength (i.e., weight) of 1.0 and the thin edges carry a strength of 0.5. Although A can activate more nodes (13) than B (10), the number of edges between the blue nodes as well as the blue nodes and B (i.e., the 15 edges in blue) is more than that between the orange nodes and the orange nodes and B (i.e., the 13 edges in orange). The total activity strength activated by B, 13, is substantially more than the total activity strength activated by A, 8.5.

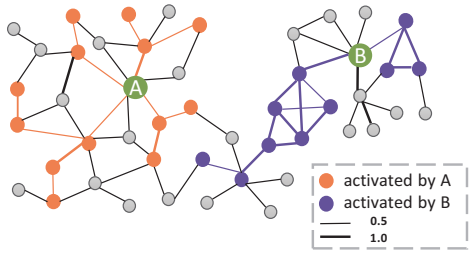


Fig. 1. A toy example showing the difference between influence maximization and activity maximization.

Activity maximization is a novel problem that is substantially different from classic influence maximization. Can we adapt some existing influence maximization methods to solve the activity maximization problem? Unfortunately, the answer is no due to the following two major reasons.

First, the activity maximization problem focuses on the interaction activities between the influenced users. This re-

- *Zhefeng Wang and Enhong Chen are with the Anhui Province Key Laboratory of Big Data Analysis and Application, School of Computer Science and Technology, University of Science and Technology of China, Hefei, China. E-mail: zhefwang@mail.ustc.edu.cn, cheneh@ustc.edu.cn (Corresponding author: Enhong Chen)*
- *Yu Yang, Lingyang Chu and Jian Pei are with the School of Computing Science, Simon Fraser University, Burnaby, Canada. E-mail: yya119@sfu.ca, lca117@sfu.ca, jpei@cs.sfu.ca*
- *Zhefeng Wang and Yu Yang contribute equally in this work*

Manuscript received Jan 26, 2017

quires comprehensive consideration of both the information diffusion dynamics and the diffusion network structure formed by the diffusion process. However, existing influence maximization methods aim to simply maximize the expected number of the active users and seldom take the diffusion network structure into consideration.

Second, at the technical level, the objective functions in the influence maximization problem and the activity maximization problem proposed here have different properties, as to be shown in Section 3. Many existing methods for the influence maximization problem rely on some special properties, such as submodurarity and supermodurarity, of the objective function in influence maximization, which unfortunately do not hold for activity maximization.

Motivated by the interesting application scenarios and the technical challenges associated, in this paper, we propose a novel problem, activity maximization, which aims to maximize the expectation of the total activity among all active users. A unique novel feature of our problem is that the optimization objective captures interactions among active users. We make several contributions.

First, we identify a novel research problem with interesting applications. We propose the novel activity maximization problem that aims to maximize the expectation of the overall activities in a social network. To the best of our knowledge, we are the first to explore the interactions among active nodes in information propagation.

Second, we assess the challenges of the proposed activity maximization problem. We show that the activity maximization problem is NP-hard under the two most popularly used information diffusion models, namely the independent cascade (IC) model and the linear threshold (LT) model. We also prove that computing the activities with respect to a given set of nodes is #P-hard under both the IC model and the LT model. Moreover, we show that the objective function of the problem is neither submodular nor supermodular. The theoretical results clearly show that the proposed activity maximization problem cannot be easily solved using the existing methods for influence maximization. To understand the feasibility of approximate solutions, we appraise the approximability of the problem by constructing a reduction from the densest k-subgraph problem.

Third, to develop practical approximate solutions, we develop a lower bound and an upper bound of activities. We prove that maximizing the lower bound or upper bound is still NP-hard under the IC model and the LT model. Moreover, computing the lower bound or upper bound is still #P-hard under the IC model and the LT model. However, we show the submodularity of the lower bound and the upper bound, which facilitates approximation.

Fourth, we develop a polling based randomized algorithm. We design a sampling method to obtain an unbiased estimation of activities. We also show how to efficiently implement the greedy strategy on the estimate of activities. We extend the sandwich approximation scheme to prove that the proposed algorithm has a data dependent approximation factor.

Last, we verify our algorithm on four real world data sets. The experimental results confirm the effectiveness and the efficiency of the proposed algorithm.

TABLE 1
Frequently used notations.

Notation	Description
$G = (V, E, B)$	A social network, where each edge $(u, v) \in E$ is associated with a diffusion model-dependent parameter $B_{u,v}$
$G_S = (V_S, E_S)$	The propagation subgraph induced by seed set S, where V_S is the set of all active nodes and $E_S = \{(u, v) \mid u \in V_S \wedge v \in V_S\}$
$n = \|V\|$	The number of nodes in G
$A_{u,v}$	The interaction strength of edge (u, v)
$\delta_A(S)$	The activity of a given seed set S
$\delta_L(\cdot)$, $\delta_U(\cdot)$	The lower bound and the upper bound respectively
g	A "live-edge" graph instance of G
$g \sim G$	g is sampled from all possible instances of G
$R_g(S)$	The set of nodes reachable from node set S in g
g^T	The transpose graph of g: $(u, v) \in g$ iff $(v, u) \in g^T$
$R_{g^T}(v)$	The reverse reachable (RR) set of node v
$\mathcal{H}$	The hypergraph consist of hyperedges
$m_{\mathcal{H}}$	The number of the hyperedges in $\mathcal{H}$
$\mathcal{D}(S)$	The degree of the node set S in $\mathcal{H}$

The rest of the pager is organized as follows. We formulate the activity maximization problem in Section 2. In Section 3, we observe several interesting and useful properties of the proposed problem. We develop a lower bound and an upper bound in Section 4. In Section 5, we devise the polling based algorithm. We review the related work in Section 6. We report the empirical evaluation results in Section 7, and conclude the paper in Section 8. Table 1 summarizes the frequently used symbols and their meanings.

2 Problem Formulation

In this section, we first review two widely used information diffusion models, and then give the formal statement of the activity maximization problem.

2.1 Diffusion Models

The independent cascade (IC) model and the linear threshold (LT) model [1] are the two most widely used information diffusion models. Our discussion in this paper is based on these two models. We briefly review them here.

Consider a social network $G = (V, E, B)$, where V is a set of vertices, $E \subseteq V \times V$ is a set of edges, and B is a diffusion model-dependent parameter. Specifically, in the IC model, $B_{u,v}$ is the propagation probability of edge (u, v), which is the probability that v is activated by u after u is activated. In the LT model, $B_{u,v}$ is the influence weight of edge (u, v), which indicates the importance of u influencing v.

Both models assume a seed set $S \subseteq V$. Let S_t be the nodes that are activated in step t $(t = 0, 1, \ldots)$ and $S_0 = S$.

In the IC model, the information diffusion process unfolds as follows. At step $t + 1$, each node v in S_t has only one chance to activate each inactive neighbor u with

the probability $B_{v,u}$. The process terminates when no more nodes can be activated.

In the LT model, the information diffusion process unfolds as follows. Initially, each node v selects a threshold θ_v in range $[0, 1]$ uniformly at random. At step $t > 0$, an inactive node v is activated if $\sum_{w \in N(v) \cap (\bigcup_{i<t} S_i)} B_{w,v} \geq \theta_v$. The process stops at a step t when $S_t = \emptyset$.

Kempe *et al.* [1] also provided an alternative perspective of the information diffusion based on "live-edge" graphs. Given a graph G, each edge is marked as "live" on certain randomized rules, and the random subgraph obtained from all live edges and all nodes in V is called the "live-edge" graph [2]. Kempe *et al.* [1] proved that we can construct equivalent "live-edge" graph models for both IC model and LT model. For the IC model, a "live-edge" graph instance can be obtained by marking each edge (u, v) as "live" with probability $B_{u,v}$ independently. For the LT model, the corresponding rule is: each node v marks at most one incoming edge (u, v) as "live" with probability $1 - \sum_{u \in N(v)} B_{u,v}$.

2.2 Activity Maximization

The activity maximization problem also considers information diffusion in a social network with an extra parameter A. Each edge $(u, v) \in E$ is associated with an activity strength $A_{u,v}$. Different from diffusion parameter $B_{u,v}$, which indicates how node u influences/activates its neighbor v, $A_{u,v}$ captures the interaction strength between u and v when they are both active. The activity strength between a pair of nodes depends on application scenarios, and take any numerical domain. For example, one may learn the activity strength from interaction log data with machine learning methods or simply use some statistical results as activity strength.

Given a social network G, an information diffusion model $\mathcal{M}$, and a seed set S, the diffusion process forms a propagation induced subgraph $G_S = (V_S, E_S)$, where V_S is the set of all active nodes and $E_S = \{(u, v) \in E \mid u \in V_S \wedge v \in V_S\}$ is the set of all edges whose two endpoints are both in V_S. Then, we can define the *activity* of a given seed set S as

$$\delta_A(S) = \mathbb{E}\left[\sum_{(u,v) \in E_S} A_{u,v}\right] \tag{1}$$

where $\mathbb{E}[\cdot]$ is the expectation operator. Since information diffusion is a stochastic process, we take the expectation with respect to all possible diffusion instances. The activity measures the overall interaction strength among the active nodes and thus can reflect the overall strength of the activity caused by the information propagated in the social network.

Now, we can formally define the activity maximization problem as follows. Given a social network G, an information diffusion model $\mathcal{M}$, and a budget k, find a seed set S^* such that

$$S^* = \arg \max_{\substack{S \subseteq V \\ |S| = k}} \delta_A(S) \tag{2}$$

From the definition, we can see that activity maximization is a discrete optimization problem, just as the traditional influence maximization problem is. Both diffusion parameter B and activity parameter A are inputs to the problem. The activity maximization problem tries to find a set of seed nodes to maximize the activity with given parameter settings. In the next section, we discuss the problem in general. Thus, the solution does no dependent on any specific settings.

3 Properties of Activity Maximization

In this section, we first prove the hardness of the activity maximization problem. Then we discuss the properties of the objective function $\delta_A(\cdot)$. Last, we show the approximability of the problem.

3.1 Hardness Results

We first assess the hardness of the activity maximization problem.

Theorem 1. *Activity maximization is NP-hard under the IC model and the LT model.*

Proof. We prove by reducing from the set cover problem [3], which is well known in NP-complete. Given a ground set $\mathcal{U} = \{u_1, u_2, \ldots, u_n\}$ and a collection of sets $\{S_1, S_2, \ldots, S_m\}$ whose union equals the ground set, the set cover problem is to decide if there exist k sets in $\mathcal{S}$ so that the union equals $\mathcal{U}$.

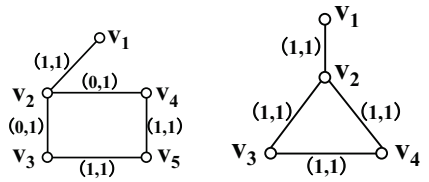
Given an instance of the set cover problem, we construct a corresponding graph with $2n + m$ nodes as follows. We create a node x_i for each set S_i, two nodes y_j and y'_j for each element u_j, and two edges (x_i, y_j) and (x_i, y'_j) with propagation probability 1 for the IC model and with influence weight 1 for the LT model and activity 0 if $u_j \in S_i$. We also create an edge between y_j and y'_j with propagation probability 0 and activity 1 for each element u_j. The information diffusion will be a deterministic process, since all propagation probabilities are either 1 or 0. Therefore, the set cover problem is equivalent to deciding if there is a set S of k nodes such that $\delta_A(S) = n$. The theorem follows immediately. □

Activity maximization is NP-hard. Then, what is the hardness of computing the activity with respect to a given seed set S?

Theorem 2. *Given a seed set S, computing $\delta_A(S)$ is #P-hard under the IC model and the LT model.*

Proof. We prove by reducing from the influence spread computation problem, which was proved #P-hard under the IC model and the LT model [4], [5].

Given an instance of the influence spread computation problem, we keep the same graph G and influence diffusion parameters B. We set $A_{u,v} = 1$ for any $u, v \in V$ and compute $x_1 = \delta_A(S)$ in the graph G. Next, we add a new node v' for each node v in the graph G and an edge between v and v' with propagation probability 1 for the IC model and with influence weight 1 for the LT model and activity 1. Now, we obtain a new graph G' and can compute $x_2 = \delta_A(S)$ in the graph G'. For any newly added node v', the only way to be activated is through its only neighbor v. Moreover, a newly added node v' will be activated if its neighbor v is active, since the propagation probability of the newly added edges is 1. Thus, $x_2 - x_1$ is exactly



(a) Counter example 1 (b) Counter example 2

Fig. 2. Counter examples

the influence spread in the graph G. The theorem follows immediately. □

In [1], Kempe *et al.* introduced the triggering model that generalizes the IC model and the LT model. In the triggering model, each node v independently chooses a subset of its neighbors as its "triggering set" according to some distribution. A node will be activated if at least one node of its triggering set is active. We can see that the reduction we construct in the proof of Theorem 2 still holds for the triggering model. Thus, we have the following result.

Corollary 2.1. *Given a seed set S, computing $\delta_A(S)$ is #P-hard in any triggering model $\mathcal{M}$ if computing influence spread is #P-hard in $\mathcal{M}$.*

3.2 Modularity of Objective Functions

The objective function of influence maximization is submodular under the IC model and the LT model. Unfortunately, the objective function in activity maximization is not submodular. Moreover, we can show that $\delta_A(\cdot)$ is not supermodular as well.

Theorem 3. *$\delta_A(\cdot)$ is not submodular under the IC model and the LT model.*

Proof. We prove by a counter example. Consider Figure 2(a). The first number in the tuple on each edge represents the propagation probability for the IC model and the influence weight for the LT model. The second number is the activity of the edge. For example, in the counter example 1, $(1, 1)$ on edge (v_1, v_2) means $B_{v_1,v_2} = 1$ and $A_{v_1,v_2} = 1$. In this example, we have $\delta_A(\{v_1\}) = 1$, $\delta_A(\{v_1, v_5\}) = 5$ and $\delta_A(\{v_5\}) = 2$. That is,

$$\delta_A(\{v_1\}) - \delta_A(\emptyset) < \delta_A(\{v_1, v_5\}) - \delta_A(\{v_5\})$$

Therefore, $\delta_A(\cdot)$ is not submodular. □

From the counter example in the proof of Theorem 3 (Figure 2(a)), we can see that the reason why $\delta_A(\cdot)$ is not submodular is the "combination effect" between the newly added node and the existing seed nodes. For example, If we add v_1 into S when $S = \emptyset$, then there is only one active endpoint for edge (v_2, v_4) and (v_2, v_3), that is v_2. But if we add v_1 to S when $S = \{v_5\}$, then both the two endpoints of edge (v_2, v_4) and (v_2, v_3) are active, since v_3 and v_4 are activated by v_5. The "combination effect" has its roots in the definition of activity. We only count the activity on the edges whose two endpoints are both active. As a result, the newly added node and the existing seed nodes may activate the two endpoints of an edge together, which leads to a violation of submodularity.

Theorem 4. *$\delta_A(\cdot)$ is not supermodular under the IC model and the LT model.*

Proof. Again, we prove by a counter example. Consider the counter example 2 in Fig 2(b), we have $\delta_A(\{v_2\}) = 4$, $\delta_A(\{v_1, v_2\}) = 4$ and $\delta_A(\{v_1\}) = 4$. Thus,

$$\delta_A(\{v_2\}) - \delta_A(\emptyset) > \delta_A(\{v_2, v_1\}) - \delta_A(\{v_1\})$$

That is, $\delta_A(\cdot)$ is not supermodular. □

From the counter example in the proof of Theorem 4 (Figure 2(b)), we can see that the reason why $\delta_A(\cdot)$ is not supermodular is the "overlap effect" between the newly added node and the existing seed nodes. The nodes that the newly added node can activate may have already been activated by the existing seed nodes, which means that adding a new node does not bring any marginal gain.

3.3 Approximability

Since $\delta_A(\cdot)$ is neither submodular nor supermodular, we cannot adopt the standard procedure for optimizing submodular function or supermodular function to get an approximation solution. To explore the approximability of the activity maximization problem, we explore the connection between the activity maximization problem and the densest k-subgraph extraction problem.

Theorem 5. *If there exists a polynomial time algorithm approximating the activity maximization problem within a ratio of α, then there exists a polynomial time algorithm that can approximate the densest k-subgraph problem within a ratio of α.*

Proof. We prove by constructing a reduction from the densest k-subgraph problem to the activity maximization problem. Given a graph and an integer k, the densest k-subgraph problem is to find a subgraph of exactly k vertices that has the maximum density. For a subgraph $G_S = (V_S, E_S)$, the density is defined as $\frac{|E_S|}{|V_S|}$.

Given an instance of the densest k-subgraph problem, we construct a corresponding instance of the activity maximization problem. We keep the same graph and set $B_{u,v} = 0$ and $A_{u,v} = 1$ for $u, v \in V$. Then, the activity maximization problem is to find a set of k vertices and maximize the number of edges whose both endpoints are in this set. It is equivalent to maximizing the density since the number of vertices is constant. □

Khot [6] showed that the densest k-subgraph problem does not admit PTAS[1] (Polynomial Time Approximation Scheme [7]) assuming $NP \nsubseteq \bigcap_{\epsilon>0} BPTIME(2^{n^\epsilon})$, we immediately have the following result.

Corollary 5.1. *There is no PTAS for the activity maximization problem assuming $NP \nsubseteq \bigcap_{\epsilon>0} BPTIME(2^{n^\epsilon})$.*

In fact, finding a good approximation to the densest k-subgraph problem is challenging. The current best approximation ratio of $n^{1/4+\epsilon}$ for $\epsilon > 0$ was achieved by Bhaskara

1. A PTAS is an algorithm that returns a solution within a factor 1 + ϵ of being optimal (or 1 - ϵ for maximization problems) in polynomial time for any $\epsilon > 0$.

et al. [8]. It is still unknown if there exists a polynomial time algorithm that can approximate the densest k-subgraph problem with a constant factor.

4 Lower Bound and Upper Bound

In this section, we first give a lower bound and an upper bound on activities. Then we discuss the properties of the lower bound and the upper bound.

4.1 The Bounds

Since the "combination effect" among seed nodes comprises the submodularity of the objective function $\delta_A(\cdot)$, we try to develop a lower bound of $\delta_A(\cdot)$ that is submodular by ignoring the "combination effect". The major idea is that we only consider the edges whose two endpoints are activated by the same seed node. Accordingly, the lower bound can be defined as

$$\delta_L(S) = \mathbb{E}[\sum_{(u,v)\in \bigcup_{x\in S} E_{\{x\}}} A_{u,v}] \tag{3}$$

where $E_{\{x\}}$ is the set of edges of the propagation subgraph induced by seed set $\{x\}$. Recall that the propagation subgraph induced by a seed set consists of the nodes that can be activated by the seed set. Here, the seed set consists of only one node x. It is easy to see that $\delta_L(S) \leq \delta_A(S)$ for any $S \subseteq V$, since we ignore the edges whose endpoints are activated by different seed nodes.

A straightforward way to get an upper bound is to consider all the edges that have at least one active endpoint. In this way, the upper bound equals to the activity of edges that have one active endpoint plus the activity of edges whose two endpoints are both active. The latter is exactly the activity we want to compute. Here, we present a tighter upper bound from the perspective of active nodes, which can be defined as

$$\delta_U(S) = \mathbb{E}[\sum_{v\in V_S} w(v)] \tag{4}$$

where

$$w(v) = \frac{1}{2}\sum_{u\in N(v)} A_{u,v}.$$

Given a seed set S, $\delta_U(S)$ equals to the half of the activity of edges that have one active endpoint plus the activity of edges whose two endpoints are both active. Thus, $\delta_U(S)$ is better than the straightforward one. Also, we can see that the upper bound is essentially a weighted version of the influence spread, where the weight of node v is $\frac{1}{2}\sum_{u\in N(v)} A_{u,v}$. For the influence spread, $w(v) = 1$ for each node v.

4.2 Properties of the Bounds

Using the lower bound and the upper bound, we can approximate the information activity problem by maximizing the lower bound and the upper bound [9]. However, maximizing the lower bound and the upper bound is still NP-hard.

Theorem 6. *Maximizing the lower bound is NP-hard under the IC model and the LT model.*

Proof. We prove by reducing from the NP-complete set cover problem [3]. We show the reduction constructed in the proof of Theorem 1 still holds for the lower bound. The lower bound only considers the edges whose two endpoints can be activated by the same seed node. In the previous reduction, for all the edges whose activity is not equal to 0 (the edges between y_j and y'_j), their two endpoints can be activated by the same node. Thus, the set cover problem can be solved by deciding if there is a set S of k nodes such that $\delta_L(S) = n$. □

Theorem 7. *Maximizing the upper bound is NP-hard under the IC model and the LT model.*

Proof. We prove by reducing from the NP-hard influence maximization problem [1].

Given an instance of the influence maximization problem, let d_{max} be the highest degree of the nodes in the graph G. Then, for each node v in G, we add $N_d = d_{max} - d_v$ new nodes, $v'_1, v'_2, \ldots, v'_{N_d}$, and N_d new edges, $(v, v'_1), (v, v'_2), \ldots, (v, v'_{N_d})$. Now we obtain a new graph G'. We set the propagation probability of the newly added edges to 0 for the IC model, and set the influence weight of the newly added edges to 0 for the LT model, and set the information activity of all the edges in G' to $\frac{2}{d_{max}}$.

Then, we have $\forall v \in V$, $w(v) = 1$, and $\forall v' \in V' \setminus V$, $w(v') = \frac{2}{d_{max}}$. Since the propagation probability of all newly added edges is 0, the newly added nodes can never be activated. Therefore, we have $I^G(S) = \delta_U^{G'}(S), \forall S \subseteq V$, where $I(S)$ is the influence spread of a give seed set S in G and $\delta_U^{G'}(S)$ is the upper bound in G'.

Next, we prove that $S_U^* = \arg\max \delta_U^{G'}(S)$ does not contain any newly added nodes. If there is any newly added node in S_U^*, we can always replace it with a node in $V \setminus S_U^*$ and increase the value of the objective function. Thus, if S_U^* is the optimal solution of maximizing the upper bound in G', it must be the optimal solution of the influence maximization in G. □

Although maximizing the lower bound and the upper bound is NP-hard, the objective functions of the lower bound and the upper bound are submodular.

Theorem 8. *$\delta_L(\cdot)$ is submodular under the IC model and the LT model.*

Proof. Given a graph G and an influence diffusion model, either the IC model or the LT model, we can construct "live-edge" graphs for G using the methods proposed in [1]. Let g be a "live-edge" graph instance. Denote by $Pr(g)$ the probability that g is selected from all possible instances. Let $E_g(S)$ be the set of edges whose two endpoints can be reachable from the same node in the seed set S. Then we can rewrite $\delta_L(S)$ to

$$\delta_L(S) = \sum_{g\sim G} Pr(g) \sum_{(u,v)\in E_g(S)} A_{u,v}$$

We only need to prove $Q(S) = \sum_{(u,v)\in E_g(S)} A_{u,v}$ is submodular for any "live-edge" graph instance g, since an non-negative linear combination of submodular functions is also submodular.

To prove, let M and N be two sets such that $M \subseteq N \subseteq V$. For any $v \in V \setminus N$, consider the difference between

$Q(M \cup \{v\})$ and $Q(M)$. It must be contributed from the edges whose two endpoints can be reachable from v but cannot be reachable from the nodes in M. These edges must be a super set of the edges whose two endpoints can be reachable from v but cannot be reachable from the nodes in N, since $M \subseteq N$. It follows that $Q(M \cup \{v\}) - Q(M) \geq Q(N \cup \{v\}) - Q(N)$. Therefore, $Q(S)$ is submodular and the theorem follows. □

Theorem 9. *$\delta_U(\cdot)$ is submodular under the IC model and the LT model.*

Proof. We can prove the theorem by the same "live-edge" technique used in the proof of Theorem 8. Let $R_g(S)$ be the set of nodes reachable from S in g. Then, $\delta_U(S)$ can be rewritten to

$$\delta_U(S) = \sum_{g \sim G} Pr(g) \sum_{v \in R_g(S)} w(v)$$

The way to prove that $Q'(S) = \sum_{v \in R_g(S)} w(v)$ is submodular is similar to the proof of $Q(S)$ in Theorem 8. The nodes that can be reachable from v but cannot be reachable from the nodes in M must be a super set of the nodes that can be reachable from v but cannot be reachable from the nodes in N. It follows that $Q'(M \cup \{v\}) - Q'(M) \geq Q'(N \cup \{v\}) - Q'(N)$. Therefore, $Q'(S)$ is submodular and the theorem follows. □

Theorems 8 and 9 are good news. With the submodularity we can adopt the standard procedure for optimizing submodular functions to obtain an approximation solution [10]. One challenge remains. Applying the algorithm proposed in [10] requires evaluating the lower bound and the upper bound. However, computing the lower bound and the upper bound with respect to a given seed set is unfortunately #P-hard.

Theorem 10. *Given a seed set S, computing $\delta_L(S)$ is #P-hard under the IC and the LT model.*

Proof. We prove by reducing from the influence spread computation problem. We show that the reduction we construct in the proof of Theorem 2 still holds for the lower bound case. Let $y_1 = \delta_L(S)$ in the graph G and $y_2 = \delta_L(S)$ in the graph G'. Since the propagation probability of the edge (v, v') is 1 for the IC model and the influence weight of the edge (v, v') is 1 for the LT model, v and v' can be activated by the same seed node. It follows that $y_2 - y_1$ is also the influence spread in the graph G. □

Theorem 11. *Given a seed set S, computing $\delta_U(S)$ is #P-hard under the IC and the LT model.*

Proof. We prove by reducing from the influence spread computation problem. The reduction is the same as the one in the proof of Theorem 7. We already showed $I^G(S) = \delta_U^{G'}(S)$ for any seed set $S \subseteq V$. Therefore, the theorem follows immediately. □

Since computing the activity, the lower bound and the upper bound is #P-hard, we will discuss how to estimate them in the next section.

5 A Polling Based Method

Recently, a polling based algorithmic framework [11], [12] was proposed for the influence maximization problem. The framework includes two steps. In the first step, it estimates the influence spread through sampling. In the second step, it finds an approximation solution for maximizing the estimate. If we can bound the estimation error, then the solution also enjoys an approximation guarantee for the influence maximization problem. To solve the activity maximization problem, we also design a polling based method.

5.1 Estimation

In a social network G, given an information diffusion model, either the IC model or the LT model, and a seed set S, let g be a "live-edge" graph instance of G and $R_g(S)$ be the set of nodes reachable from S in g. Denote by $R_{g^T}(v)$ the reverse reachable (RR) set [13] for node v in g, where g^T is the transpose graph [11] of g: $(u, v) \in g$ iff $(v, u) \in g^T$. We write $(u, v) \sim E$ to indicate that we randomly pick (u, v) from E as a sample according to a certain distribution. The meaning of $v \sim V$ is similar.

To estimate the activity, we first have the following result.

Theorem 12. *For any seed set $S \subseteq V$,*

$$\delta_A(S) = T \cdot \Pr_{g \sim G, (u,v) \sim E}\Big[S \cap R_{g^T}(u) \neq \emptyset \wedge S \cap R_{g^T}(v) \neq \emptyset\Big],$$

where $T = \sum_{(u,v) \in E} A_{u,v}$.

Proof.

$$\begin{aligned}
\delta_A(S) &= \mathbb{E}\Big[\sum_{(u,v) \in E_S} A_{u,v}\Big] \\
&= \sum_{(u,v) \in E} Pr\Big[(u,v) \in E_S\Big] A_{u,v} \\
&= \sum_{(u,v) \in E} \Pr_{g \sim G}\Big[u \in R_g(S) \wedge v \in R_g(S)\Big] A_{u,v} \\
&= \sum_{(u,v) \in E} \Pr_{g \sim G}\Big[\exists w_1, w_2 \in S, w_1 \in R_{g^T}(u) \wedge \\
&\qquad w_2 \in R_{g^T}(v)\Big] A_{u,v} \\
&= T \cdot \sum_{(u,v) \in E} \Pr_{g \sim G}\Big[\exists w_1, w_2 \in S, w_1 \in R_{g^T}(u) \wedge \\
&\qquad w_2 \in R_{g^T}(v)\Big] \frac{A_{u,v}}{T} \qquad (5) \\
&= T \cdot \Pr_{g \sim G, (u,v) \sim E}\Big[\exists w_1, w_2 \in S, w_1 \in R_{g^T}(u) \wedge \\
&\qquad w_2 \in R_{g^T}(v)\Big] \\
&= T \cdot \Pr_{g \sim G, (u,v) \sim E}\Big[S \cap R_{g^T}(u) \neq \emptyset \wedge S \cap R_{g^T}(v) \neq \emptyset\Big]
\end{aligned}$$

Eq. 5 is the expected probability with respect to the activity distribution of edges, where the probability for edge (u, v) is $\frac{A_{u,v}}{T}$. □

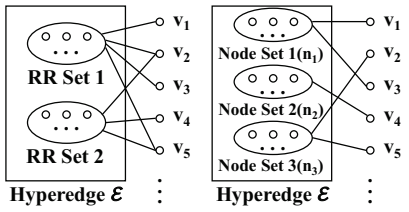


Fig. 3. Hyperedge for activities

Input: Social network $G = (V, E, B)$, A and diffusion model $\mathcal{M}$
Output: A hyperedge $\mathcal{E}$
1: Initialize $\mathcal{E} = (\emptyset, \emptyset)$
2: Pick an edge (u, v) with probability $\frac{A_{u,v}}{T}$.
3: Generate a "live-edge" graph g according to $\mathcal{M}$
4: Let $N_1 = R_{g^T}(u)$ and $N_2 = R_{g^T}(v)$
5: Let $\mathcal{E} = (N_1, N_2)$
6: **return** $\mathcal{E}$

Algorithm 1: Generate Hyperedges

The intuition of Theorem 12 is that if a seed set S has a high activity value, then the probability that S is simultaneously reachable from both two endpoints of a randomly picked edge in a randomly picked "live-edge" graph instance is high, since we only count the activity on the edges whose two endpoints are both active. Theorem 12 implies that we can estimate $\delta_A(S)$ by estimating the probability of the event $S \cap R_{g^T}(u) \neq \emptyset \wedge S \cap R_{g^T}(v) \neq \emptyset$. To achieve the estimation, we conduct a poll as follows. We select an edge (u, v) with probability $\frac{A_{u,v}}{T}$, and run Monte Carlo simulation of the "live-edge" process. During the process, we record all the nodes that can reach u and v through "live" edges. Algorithm 1 summarizes the process.

One critical observation is that we do not need to conduct the "live-edge" process on the entire graph. Instead, we can simulate the process starting from u and v, respectively. We only need to make sure that each edge is marked consistently as the same status ("live" or "blocked") in these two simulations. We call the pair of two RR sets obtained from a poll a hyperedge. All the generated hyperedges constitute a hypergraph $\mathcal{H}$.

Denote by m_H the number of the hyperedges in $\mathcal{H}$. If a node v appears in both RR sets of a hyperedge $\mathcal{E}$, $\mathcal{E}$ is said to be *fully covered* by v. If a node v only appears in one of the two RR sets of a hyperedge $\mathcal{E}$, $\mathcal{E}$ is said to be *partially covered* by v. Denote by $\mathcal{D}(S)$ the degree of the set of nodes S, which is the number of hyperedges in $\mathcal{H}$ that can be fully covered by S. According to Theorem 12, $T \cdot \frac{\mathcal{D}(S)}{m_H}$ is an unbiased estimator of $\delta_A(S)$ for any fixed m_H. Please note that there also exists "combination effect" between nodes in this case. For example, in the left part of Figure 3, v_1 only appears in the first RR set of hyperedge $\mathcal{E}$ and v_4 only appears in the second RR set. v_1 and v_4, respectively, partially covers $\mathcal{E}$. But $\mathcal{E}$ is fully covered by the combination of v_1 and v_4. Thus, similar to $\delta_A(\cdot)$, $\mathcal{D}(\cdot)$ is not submodular neither.

Similarly, for the lower bound and the upper bound, we have the following two results.

Theorem 13. *For any seed set $S \subseteq V$,*

$$\delta_L(S) = T \cdot \Pr_{g\sim G,(u,v)\sim E}\Big[S \cap (R_{g^T}(u) \cap R_{g^T}(v)) \neq \emptyset\Big],$$

where $T = \sum_{(u,v)\in E} A_{u,v}$.

Proof. The lower bound only considers the edges whose two endpoints can be activated by the same seed node. Thus, to prove the theorem, we only need to let $w_1 = w_2$ in the proof of Theorem 12, that is

$$\begin{aligned}\delta_L(S) =& T \cdot \Pr_{g\sim G,(u,v)\sim E}\Big[\exists w \in S, w \in R_{g^T}(u) \wedge w \in R_{g^T}(v)\Big] \\ =& T \cdot \Pr_{g\sim G,(u,v)\sim E}\Big[S \cap (R_{g^T}(u) \cap R_{g^T}(v)) \neq \emptyset\Big]\end{aligned}$$

□

Using Theorem 13, we can estimate the lower bound using essentially the same sampling process as the activity. The only difference is that there is only one node set in the hyperedge for the lower bound, that is $N_1 \cap N_2$. In this case, a hyperedge $\mathcal{E}$ is covered by node v if and only if $v \in N_1 \cap N_2$.

Theorem 14. *For any seed set $S \subseteq V$,*

$$\delta_U(S) = W \cdot \Pr_{g\sim G,v\sim V}\Big[S \cap R_{g^T}(v) \neq \emptyset\Big],$$

where $W = \sum_{v\in V} w(v)$.

Proof. The upper bound is essentially a weighted variation of the influence spread. Thus, we can apply the proof proposed in [14]. □

There is also only one node set in the hyperedge for the upper bound. We can generate the hyperedge using the sampling method proposed in [14].

Since we can estimate the objective function ($\delta_A(\cdot)$, $\delta_L(\cdot)$ or $\delta_U(\cdot)$) by the degrees of the set of nodes, we can regard $\mathcal{H}$ as encoding an approximation to the objective function. With the estimate of the objective function, we go to the second step of the polling based framework, that is, maximizing the estimate. To achieve this goal, we adopt the simple but powerful greedy strategy, which picks the node with the largest marginal gain (the increase of degree in $\mathcal{H}$ for our case) iteratively. Next, we show how to efficiently implement a greedy strategy on the hypergraph.

5.2 Efficient Implementation of the Greedy Strategy

For the lower bound and the upper bound, there is only one node set in each hyperedge. Thus, we can use the standard greedy algorithm for maximum coverage problem to obtain an approximate solution [13]. However, there are two node sets in the hyperedge for the activity maximization problem. A hyperedge can be fully or partially covered by a node or a node set. Thus, we cannot directly apply the greedy strategy. To tackle this issue, here we discuss how to efficiently implement the greedy strategy on the hypergraph.

First, we store the original hyperedges of two RR sets in a more efficient manner. There are three sets, n_1, n_2 and n_3 for each hyperedge $\mathcal{E}$, where n_1 and n_2 are the sets of nodes that can only cover the first and second RR set of $\mathcal{E}$, respectively, and n_3 is the set of nodes that can cover both two RR sets of $\mathcal{E}$. Figure 3 illustrates the idea.

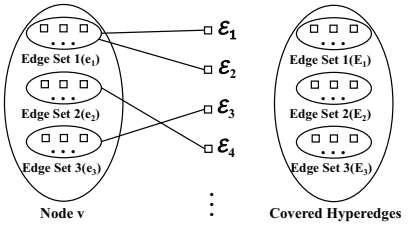


Fig. 4. Data Structures

Then, we build an inverted index for each node. There are three sets, e_1, e_2 and e_3 for each node v, where e_1 and e_2 are the sets of hyperedges whose first and second RR set can be covered by v, respectively, and e_3 is the set of hyperedges that can be fully covered by v.

Third, we maintain a global data structure to record the current covered hyperedges. There are also three sets, E_1, E_2 and E_3, in this data structure, where E_1 and E_2 are the sets of hyperedges whose first and second RR sets have been covered, respectively, and E_3 is the set of hyperedges that have been fully covered. Figure 4 shows these two data structures. With these data structures, we have the following fact.

Fact 1. *Given a seed set S, for each vertex $v \in V \setminus S$, the marginal gain $\mathcal{D}(S \cup \{v\}) - \mathcal{D}(S)$ is*

$$MG(v) = |v.e_3 \setminus E_3| + |v.e_1 \cap E_2| + |v.e_2 \cap E_1| \quad (6)$$

RATIONALE. If we add a node v to the current seed set S, the newly covered hyperedges can be divided into two groups. The first group is the hyperedges that can be covered by v alone but not covered by S, that is $v.e_3 \setminus E_3$. The second group is the hyperedges that are partially covered by S and are fully covered if v is added to S, that is $v.e_1 \cap E_2$ and $v.e_2 \cap E_1$. ■

Fact 1 implies that we can pick the node with the largest marginal gain in each iteration and then incrementally update the marginal gains of the rest nodes. Algorithm 2 describes the details.

Here, we briefly explain how to incrementally update the marginal gain. Assuming E_1, E_2 and E_3 are updated to E_1', E_2', and E_3', respectively, we update the marginal gains as follows. For each hyperedge $\mathcal{E} \in E_1' \setminus E_1$, we increase the marginal gains of the nodes in $\mathcal{E}.n_2$ by 1. For each hyperedge $\mathcal{E} \in E_2' \setminus E_2$, we increase the marginal gains of the nodes in $\mathcal{E}.n_1$ by 1. For each hyperedge $\mathcal{E} \in E_3' \setminus E_3$, we first decrease the marginal gains of the nodes in $\mathcal{E}.n_3$ by 1. Then, we decrease the marginal gains of the nodes in $\mathcal{E}.n_2$ by 1 if $\mathcal{E} \in E_1$, and decrease the marginal gains of the nodes in $\mathcal{E}.n_1$ by 1 if $\mathcal{E} \in E_2$.

Now, the only remaining question is to decide how many hyperedges we need to sample, which will be addressed next.

5.3 Sample Complexity

In this subsection, we discuss how to use a sample of proper size to restrict the estimate error of the activity, the lower bound and the upper bound. With the technique, we show that the polling algorithm can provide an approximate solution to maximizing the lower bound and the upper bound.

Input: Social network G, Hypergraph $\mathcal{H}$ and budget k
Output: Seed set S

```
Initialize S = E1 = E2 = E3 = ∅
for v ∈ V do
    MG(v) = |v.e|
end for
while |S| < k do
    v = arg max_{u∈V\S} MG(u)
    S = S ∪ {v}
    update E1, E2, and E3
    for u ∈ V \ S do
        update MG(u) according to Eq. 6
    end for
end while
return S
```

Algorithm 2: Maximum Coverage on Hypergraph

To bound the estimate error of the polling method, we have the following lemma from [15].

Lemma 1. *Let $Z_1, Z_2, \ldots$ be independently and identically distributed according to Z in the interval $[0, 1]$ with mean μ_Z. Let $S = \sum_{i=0}^{N} Z_N$ and $\hat{\mu}_Z = \frac{S}{N}$. Let $\Upsilon = 4(e-2)\frac{\ln(2/\delta)}{\epsilon^2}$ and $\Upsilon_1 = 1 + (1+\epsilon)\Upsilon$. If N is the number of samples when $S >= \Upsilon_1$, then $Pr[|\hat{\mu}_Z - \mu_Z| \leq \epsilon\mu_Z] > 1 - \delta$ and $\mathbb{E}[N] \leq \Upsilon_1/\mu_Z$.*

Lemma 1 provides a stopping condition for the sampling process. Given a seed set S, we can keep sampling hyperedges until $\mathcal{D}(S) \geq \Upsilon_1$. Then, $T \cdot \frac{\mathcal{D}(S)}{m_H}$ is an (ϵ, δ) estimation [16] of $\delta_A(S)$. The analysis is similar in the cases of the lower bound and the upper bound.

Nguyen *et al.* [17] analyzed the conditions that the polling algorithmic framework must meet to obtain an approximation solution. Let S^* be the optimal seed set and $\hat{S}$ be the seed set returned by the greedy strategy on the estimate of the objective function $f(\cdot)$ ($\delta_L(\cdot)$ or $\delta_U(\cdot)$). Denote by $\hat{f}(\cdot)$ the estimate of the objective function $f(\cdot)$. The conditions are

$$Pr[\hat{f}(\hat{S}) \leq (1+\epsilon_1) f(\hat{S})] \geq 1 - \delta_1 \quad (7)$$

$$Pr[\hat{f}(S^*) \geq (1-\epsilon_2) f(S^*)] \geq 1 - \delta_2 \quad (8)$$

where $\delta_1 + \delta_2 \leq \delta$ and $\epsilon_1 + (1 - 1/e)\epsilon_2 \leq \epsilon$. Let N be the number of samples such that both Eq. 7 and Eq. 8 are guaranteed. Then we have the following lemma from [17].

Lemma 2. *Given a social network G, if the number of hyperedges $m_H \geq N$, then the polling algorithm returns $\hat{S}$ satisfying $Pr[f(\hat{S}) \geq (1-1/e-\epsilon) f(S^*)] \geq 1-\delta$ and $\hat{S}$ is an $(1-1/e-\epsilon)$ approximate solution.*

Using Lemmas 1 and 2, to obtain an approximation solution to maximizing the lower bound or upper bound, we can keep sampling hyperedges and checking if the conditions are met. SSA algorithm from [17] describes the process.

Using SSA algorithm, we can provide a $(1 - 1/e - \epsilon)$ approximation solution to maximizing the lower bound and the upper bound with probability of at least $1 - \delta$. But

1: Let S_U be a α approximation to the upper bound
2: Let S_L be a β approximation to the lower bound
3: Let S_A be a solution to the original problem
4: $\hat{\delta}_A(\cdot)$ is a multiplicative γ-error estimate of $\delta_A(\cdot)$
5: $S = \arg\max_{S_0 \in \{S_U, S_L, S_A\}} \hat{\delta}_A(S_0)$
6: **return** S

Algorithm 3: Sandwich Approximation Framework

we must point out that the analysis does not hold for the activity maximization problem. This is because a necessary condition of the polling algorithmic framework is that we can approximate the estimate using the greedy strategy. The condition is not met in the case of the activity maximization problem, since the estimate of the activity is not submodular. Thus, the polling algorithm cannot provide an approximation solution to the activity maximization problem. But it is still a good heuristic for the activity maximization problem. Furthermore, by combining the approximation algorithm for the lower bound and the upper bound, we can derive a data dependent approximation scheme for the activity maximization problem.

5.4 Data Dependent Approximation

There is no general way to optimize or approximate a non-submodular function. Lu *et al.* [9] proposed a sandwich approximation strategy, which approximates the objective function by approximating its lower bound and upper bound. The sandwich approximation strategy works as follows. First, we find a solution to the original problem with any strategy. Second, we find an approximate solution to the lower bound and the upper bound, respectively. Last, we return the solution that has the best result for the original problem.

Here, we extend the strategy to the case in which the objective function is intractable and have the following result.

Theorem 15. *Let S be the seed set returned by Algorithm 3, then we have*

$$\delta_A(S) \geq \max\left\{\frac{\delta_A(S_U)}{\delta_U(S_U)}\alpha, \frac{\delta_L(S_L^*)}{\delta_A(S_A^*)}\beta\right\}\frac{1-\gamma}{1+\gamma}\delta_A(S_A^*) \quad (9)$$

Proof. Let S_L^*, S_U^* and S_A^* be the optimal solutions to maximizing the lower bound, the upper bound and the activity, respectively. Then, we have

$$\begin{aligned}\delta_A(S_U) &= \frac{\delta_A(S_U)}{\delta_U(S_U)}\delta_U(S_U) \geq \frac{\delta_A(S_U)}{\delta_U(S_U)} \cdot \alpha \cdot \delta_U(S_U^*) \\ &\geq \frac{\delta_A(S_U)}{\delta_U(S_U)} \cdot \alpha \cdot \delta_U(S_A^*) \geq \frac{\delta_A(S_U)}{\delta_U(S_U)} \cdot \alpha \cdot \delta_A(S_A^*) \\ \delta_A(S_L) &\geq \delta_L(S_L) \geq \beta \cdot \delta_L(S_L^*) \geq \frac{\delta_L(S_L^*)}{\delta_A(S_A^*)} \cdot \beta \cdot \delta_A(S_A^*)\end{aligned}$$

Let $S_{max} = \arg\max_{S_0 \in \{S_U, S_L, S_A\}} \delta_A(S_0)$, then

$$\delta_A(S_{max}) \geq \max\left\{\frac{\delta_A(S_U)}{\delta_U(S_U)}\alpha, \frac{\delta_L(S_L^*)}{\delta_A(S_A^*)}\beta\right\}\delta_A(S_A^*)$$

Since $\forall S_0 \in \{S_U, S_L, S_A\}, |\hat{\delta}_A(S_0) - \delta_A(S_0)| \leq \gamma\delta_A(S_0)$, we have $(1+\gamma)\delta_A(S) \geq (1-\gamma)\delta_A(S_{max})$. It follows that

$$\delta_A(S) \geq \frac{(1-\gamma)}{(1+\gamma)}\delta_A(S_{max})$$

□

Theorem 15 indicates that we can approximate the activity maximization problem within a factor that is dependent on the data. Since it is #P-hard to compute $\delta_A(\cdot)$ and $\delta_U(\cdot)$, and is NP-hard to find S_L^* and S_A^*, we cannot compute the exact approximation factor. But we can estimate $\frac{\delta_A(S_U)}{\delta_U(S_U)}$ by computing its lower bound $\frac{(1-\gamma)\hat{\delta}_A(S_U)}{(1+\gamma)\hat{\delta}_U(S_U)}$. It follows that $\frac{(1-\gamma)^2}{(1+\gamma)^2} \cdot \alpha \cdot \frac{\hat{\delta}_A(S_U)}{\hat{\delta}_U(S_U)}$ is a computable lower bound of the approximation factor.

Now, we put all the pieces of the puzzle together. We first adopt the polling algorithm to maximize the lower bound and the upper bound. As discussed in Section 5.3, it provides $(1-1/e-\epsilon)$ approximate solutions to the lower bound and the upper bound, respectively. Consequently, we have $\alpha = \beta = (1 - \frac{1}{e} - \epsilon)$ in Algorithm 3. Then, we also use the polling algorithm to get a heuristic solution (S_A) to the activity maximization problem. Last, we get a (γ, δ) estimation of $\delta_A(\cdot)$ based on Lemma 1 to complete Line 5 of Algorithm 3. According to Theorem 15, the sandwich algorithm returns a seed set S such that

$$\delta_A(S) \geq \max\left\{\frac{\delta_A(S_U)}{\delta_U(S_U)}, \frac{\delta_L(S_L^*)}{\delta_A(S_A^*)}\right\}\frac{1-\gamma}{1+\gamma}(1 - \frac{1}{e} - \epsilon)\delta_A(S_A^*)$$

6 Related Work

Domingos and Richardson [18] first exploited the influence between users in social networks for viral marketing. The key idea behind viral marketing is that, by targeting on only a small number of individuals (in practice, for example, persuading them to adopt the product), we can trigger a large cascade of (product) adoption spreading in a social network. Kempe *et al.* [1] formulated the problem as a discrete optimization problem, which is also well known as the influence maximization problem. The influence maximization problem aims to optimize the influence spread (the expected number of active nodes) in a given information diffusion model, such as the IC model and the LT model. Due to its important applications in viral marketing and some other areas, it has drawn much attention from both academia and industry [19], [20], [21], [22], [23], [24], [25]. Influence maximization is good for the scenarios of viral marketing, because the active state of a node means product adoption. But for topic promotion, the activity of a topic also depends on the interactions among active nodes, which are exactly where activity maximization differs from influence maximization.

Under the IC model and the LT model, Kempe *et al.* [1] proved that influence maximization is NP-hard. Moreover, Chen *et al.* [4], [5] proved that computing influence spread is #P-hard. Thus, many heuristic algorithms were proposed to solve the problem under these two models [4], [5], [26], [27], [28], [29]. Recently, a polling based method [11] was proposed for influence maximization. Unlike the previous heuristic algorithms, this method can provide a solution with provable approximation guarantee. Later, Tang *et al.* [12], [13] reduced the sample complexity and improved the efficiency. Nguyen *et al.* [17] further sped up the algorithm with a different bounding technique [15]. In this paper,

TABLE 2
The statistics of the data sets.

Network	# Vertices	# Edges	Average degree
Douban	45,559	293,377	6.4
Aminer	1,712,433	4,258,615	2.5
DBLP	317,080	1,049,866	3.3
LiveJournal	3,997,962	34,681,189	8.7

we extend this algorithmic framework to solve our activity maximization problem in a non-trivial way.

Although influence maximization has its root in viral marketing, it may still be impractical under many real-life scenarios. To fill this gap, a series of extensions to the influence maximization problem were studied. For example, Goyal *et al.* [30] proposed a data based approach to influence maximization based on a credit distribution model. Instead of maximizing the influence spread under some propagation models with respect to some learned parameters, they tried to find influential nodes from the action log data directly. Chen *et al.* [31] considered the time-delay aspect of influence diffusion and studied the influence maximization with time-critical constraint. Similarly, the spatial factor of influence diffusion is considered and influence maximization on Euclidean space has been studied as well [32], [33], [34]. Tang *et al.* [35] studied the problem of maximizing the influence spread and the diversity of the influenced crowd simultaneously. Bhagat *et al.* [36] argued that product adoption should be distinguished from influence spread in viral marketing, as influence spread is essentially used as "proxy" for product adoption. Wang *et al.* [37] distinguished the information coverage and information propagation, and proposed a new optimization objective that includes the values of the informed nodes. All these extensions were from the perspective of nodes and tried to exploit the values of nodes as separate individuals in different diffusion models and different problem settings. They did not consider activity strengths on edges in their objectives. In contrast, our problem captures the interactions among nodes and enables different (often orthogonal) applications of information diffusion.

7 Experiments

In this section, we evaluate our algorithm via a series of experiments on four real-world data sets.

7.1 Settings

We ran our experiments on four real-world data sets, which include **Douban** [38], **AMiner** [39], **DBLP** [40] and **LiveJournal** [40]. The last two data sets are available at the SNAP website (http://snap.stanford.edu). Table 2 shows the statistics of the data sets.

The Douban data set is a social network about movie ratings. We use the number of shared movies between a pair of users as their activity strength, which reflects the common interest between users. The AMiner data set is an academic social network. We use the number of co-authored papers between a pair of users as their activity strength, which reflects the collaboration strength between users. To explore the possibility of other kind of activity strengths, we also verify our algorithm using two synthetic activity settings on the DBLP data set and the Livejournal data set. In the first case, we uniformly set $A_{u,v}$ to 1 for each edge (u, v). In the second case, we set $A_{u,v}$ to the value of the diffusion parameter $B_{u,v}$. The intuition is that there may be more interactions between u and v if u is more likely to activate v. The propagation probability $B_{u,v}$ for the IC model and the influence weight for the LT model of an edge (u, v) is set to $\frac{1}{degree(v)}$, as widely used in literature [2]. For the parameters controling approximation quality, we set $\epsilon = 0.1$, $\delta = 0.001$ and $\gamma = 0.05$ for all data sets.

We compare the proposed algorithm, referred as Sandwich, with three heuristic algorithms: InfMax, Degree and PageRank. InfMax returns the nodes for influence maximization. We followed the implementation reported in [17]. Degree returns the nodes with high degrees. PageRank returns the nodes with high PageRank [41] scores.

We implemented our algorithm and the baselines in Java. All experiments were conducted on a PC with a 3.4GHZ Intel Core i7-3770 processor and 32 GB memory, running Microsoft Windows 7.

7.2 Effectiveness

Figure 5 shows the activity computed by each algorithm on the four data sets, respectively. For better illustration, we report the *comparative gain ratio* instead of the absolute activity value, since the activity value scales vary greatly with respect to seed set size. It is easier to distinguish the gaps between the baselines and our algorithm when we use comparative gain ratio as the metric, since it is not affected by activity value scales. The comparative gain ratio of an algorithm $\mathcal{A}$ is defined as $\frac{\delta_A(S_{\mathcal{A}})-\delta_A(S)}{\delta_A(S)}$, where $S_{\mathcal{A}}$ and S are the seed sets returned by algorithm $\mathcal{A}$ and the Sandwich algorithm, respectively.

In the cases of real activity settings, our algorithm Sandwich always has the best performance. Only in very few cases of synthetic activity settings, Sandwich is outperformed marginally. In the real activity settings, InfMax has a poor performance and almost in all cases is the poorest one. This is because influence maximization only considers the number of active nodes and ignores the network structure. This phenomenon also demonstrates what we have pointed out in Section 1: more active users do not necessarily lead to more interaction activities.

In the uniform settings, algorithm Degree performs well under the IC model but has a relatively bad performance under the LT model. InfMax and PageRank often have a bad performance under both the IC model and the LT model in the uniform settings. In the diffusion settings, InfMax algorithm is a good heuristic under both the IC model and the LT model. Algorithm PageRank performs well on the DBLP data set but has a bad performance on the other two data sets. Algorithm Degree often has a bad performance under both the IC model and the LT model in the diffusion settings. These results show that these baseline algorithms are not stable in performance in this task and can only work well in some specific data set or activity setting. The reason is that these baseline algorithms only use the properties of the social network or the diffusion process but totally ignore the activity strengths on edges. In contrast, our algorithm

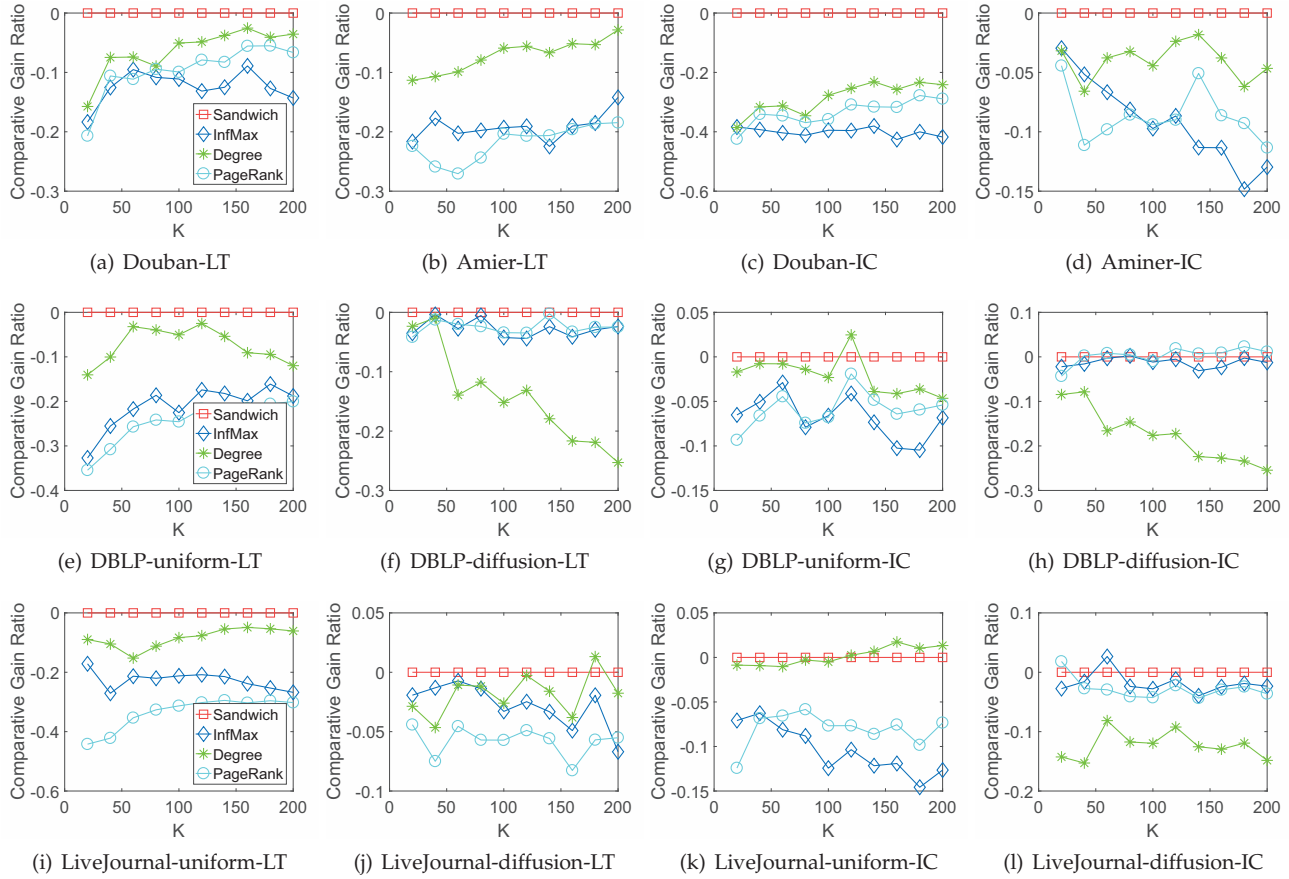


Fig. 5. Information activity on four data sets. (a)-(d) show the performances on two real data sets (Douban and Aminer) under IC and LT models. (e)-(l) show the performances on real data sets (DBLP and LiveJournal) with two types of synthetic activity settings, such as uniform and diffusion.

utilizes the unbiased estimate of the activity and its lower and upper bounds to solve the problem. This is why our algorithm always has a good performance while the baseline algorithms fail in many cases.

7.3 Approximation Quality

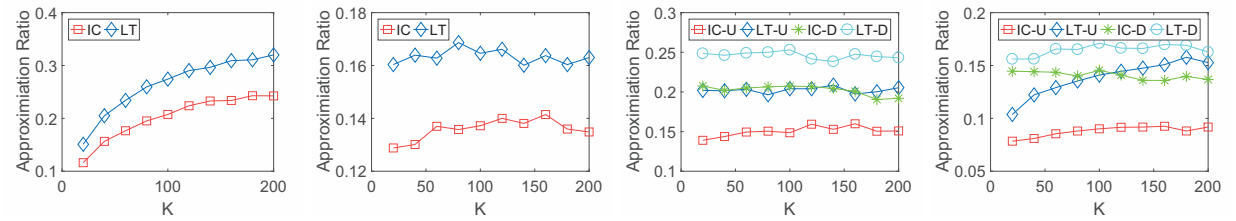
A major advantage of our algorithm is that it carries a data dependent approximation ratio. Since the exact approximation is intractable to compute, we report the computable lower bound of the approximation ratio, that is $\frac{(1-\gamma)^2}{(1+\gamma)^2} \cdot (1-e-\epsilon) \cdot \frac{\hat{\delta}_A(S_U)}{\hat{\delta}_U(S_U)}$. Figure 6 shows the results on the four data sets.

The ratio varies in different data sets. On the same data set, the ratios under the IC model and the LT model also differ. In general, the ratio under the LT model is greater than the one under the IC model in the same activity settings. The ratio does not change much with respect to the size of the seed set k. Roughly the ratio increases when k increases. A possible reason is that the gap between the activity and the upper bound shrinks when k increases, since there are more nodes activated with a larger value of k. Interestingly, we observe that, in terms of approximation ratio, the LT model consistently outperforms the IC model on both the real data sets (i.e., Douban and Aminer in Figures 6(a)-(b)) and the data sets with synthetic activities (i.e., DBLP and LiveJournal in Figures 6(c)-(d)). The consistency in the experimental results indicates that the uniform setting and diffusion setting of synthetic activities are two possible ways to simulate real activities.

7.4 Scalability

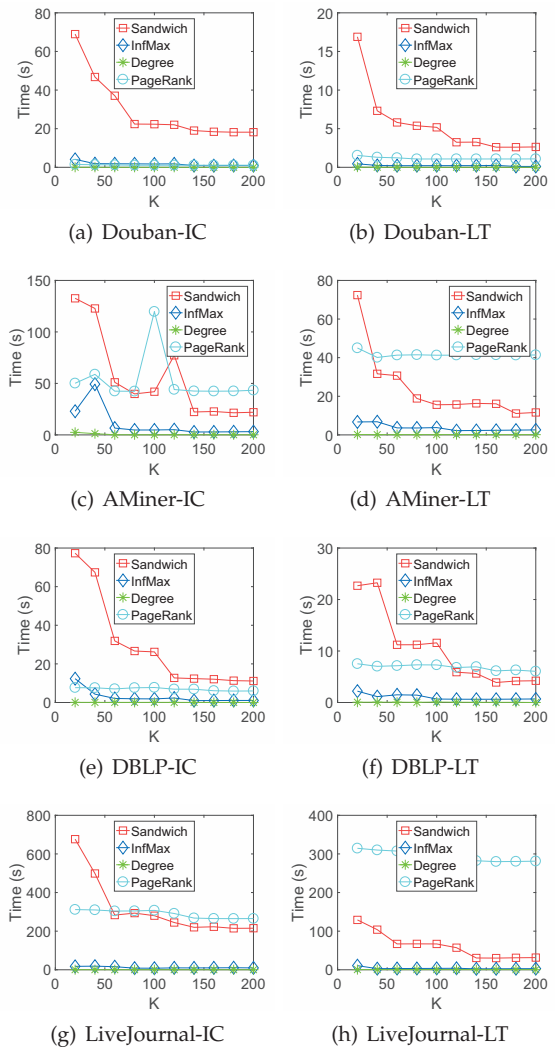
Since the activity settings do not affect the running time, we only report the running time in the uniform case. Figure 7 shows the running time on the four data sets.

In most of the cases, the running time of our algorithm decreases when the size of seed set k increases. This is because the time cost in Sandwich depends on the number of sampled hyperedges. According to Lemma 1, the expected number of samples is inversely proportional to μ_Z, which is the probability of the event $S \cap R_{g^T}(u) \neq \emptyset \wedge S \cap R_{g^T}(v) \neq \emptyset$. It increases when k increases. A similar analysis holds for the lower bound and the upper bound. PageRank is faster than our algorithm on the smallest data sets but slower on the largest data set. Degree and InfMax are more efficient than our algorithm, but they are substantially weaker than ours in effectiveness in many cases. As described in the previous sections, there are many differences between our algorithm and InfMax, which lead to different time costs of the two algorithms. First, to obtain a data dependent approximation factor, the Sandwich algorithm actually solves three problems with polling based method. Second, during the sampling process of the original problem and maximizing the lower bound, we need to conduct a poll from both two endpoints of a randomly picked edge. Third,



(a) Douban (b) Aminer (c) DBLP (d) LiveJournal

Fig. 6. The performance of approximation ratio on four data sets. (a)-(b) show the performances on Douban and Aminer under IC model and LT model. (c)-(d) show the performances on DBLP and LiveJournal with two types of synthetic activity settings, such as uniform (U) and diffusion (D).



(a) Douban-IC (b) Douban-LT

(c) AMiner-IC (d) AMiner-LT

(e) DBLP-IC (f) DBLP-LT

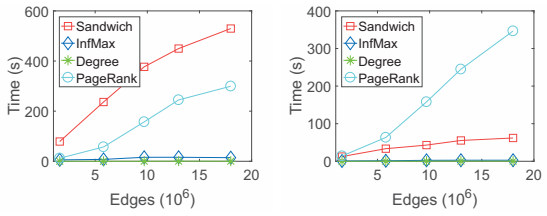
(g) LiveJournal-IC (h) LiveJournal-LT

Fig. 7. The running time of all methods on four data sets under IC model and LT model. We only report the running time under the uniform activity setting, because the activity settings do not affect the running time.

TABLE 3
The sampled LiveJournal Data sets

Sample ID	1	2	3	4	5
Vertices ($\times 10^5$)	1.0	5.0	9.0	13	20
Edges ($\times 10^6$)	1.6	5.8	9.7	13	18

To further explore the scalability of the algorithms, we sample five data sets from the LiveJournal data sets as follows. First, we start a breadth first search (BFS) from a randomly selected node on the whole graph G until the desired number of nodes are visited. Denote by N the set of all nodes visited by the BFS. We use N to induce a subgraph G_N as the sampled data set. The number of nodes and edges of the five subgraphs are listed in Table 3. After we obtain the sample data sets, we run the algorithms when the size of seed set is set to 200. The results are shown in Figure 8. For both the IC model and the LT model, the Sandwich algorithm scales up roughly linearly with respect to the number of edges. Also, the slope in the IC model is greater than that in the LT model. In other words, the time cost increases more rapidly in the IC model. A possible reason is that the influence in the IC model is more sensitive to the number of edges in the graph, since each edge is activated independently in the IC model.



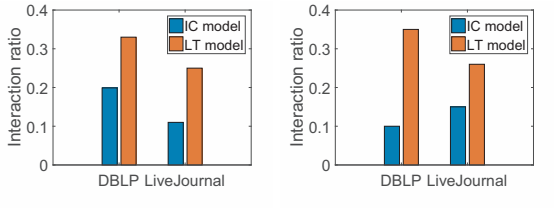
(a) IC model (b) LT model

Fig. 8. The run running time of all methods on five sampled data sets

the sampling objects of the two algorithms are different, i.e., nodes versus edges. The sampling complexity of InfMax is mainly dependent on the number of nodes while the sampling complexity of Sandwich is mainly dependent on the number of edges. It is worthy noting that our algorithm is actually very efficient. The largest running time is only about 600 seconds on the largest data set, which has millions of nodes and tens of millions of edges.

7.5 Influence Spread versus Activity

To explore the relation between influence spread and activity strength, we report their values on the DBLP data set and the LiveJournal data set with the uniform settings. We choose the uniform settings for these experiments here because, in such a situation, the total nactivity strength is exactly the number of edges between the active nodes. In this case, the activities can reflect the effect of the network structure formed by the propagation induced subgraph. In the the other two data sets, there is no such correspondence.



(a) $k = 20$ (b) $k = 200$

Fig. 9. The interaction ratio performances of IC model and LT model on DBLP and LiveJournal data sets under uniform activity setting.

We also calculate the ratio, which is the influence spread against the information activity. Table 4 shows the results on the two data sets.

The ratio differs under different models. In general, the ratio under the LT model is greater than the one under the IC model. Possibly active nodes are more closely connected to each other under the LT model. We also notice that the ratio is similar when $k = 20$ and $k = 200$. This result suggests that the relation between the influence spread and the activity does not vary much with respect to the size of seed set. The reason is that more seed nodes lead to more active nodes, but, at the same time, the activity also depends on the network structure among these nodes.

The ratio can be viewed as the average degree of the propagation induced subgraph. The average degree of the propagation induced subgraph is smaller than the average degree of the whole graph. This is because only a small proportion of the nodes can be activated. Thus, there are many edges between active nodes and inactive nodes. The average degree of the propagation induced subgraph only considers the edges between active nodes. Thus, we report the interaction ratio of the active nodes, which is the number of edges whose both endpoints are active against the number of edges that have at least one active endpoint. The results are shown in Figure 9. The interaction ratios are not high on the two data sets. This indicates that only a small proportion of the neighbors are activated and interact with the active nodes. This result demonstrates an essential difference between activity maximization and influence maximization.

8 Conclusions

In this paper, to address the demand raised in several interesting applications, we proposed and formulated a novel problem, activity maximization. We proved the hardness of the problem under both the IC model and the LT model. We also developed a lower bound and an upper bound of the objective function, and observed several useful properties of the lower bound and the upper bound. We designed a polling based algorithm to solve the problem that carries a data dependent approximation ratio. Our experimental results on four real data sets verified the effectiveness and efficiency of our method. As future work we are interested in learning the activity of user pairs from real-world data.

Acknowledgement

This work was partially supported by grants from the National Natural Science Foundation of China (Grants No. U1605251, 61325010 and 61672483), the NSERC Discovery Grant program, the Canada Research Chair program, and the NSERC Strategic Grant program. All opinions, findings, conclusions and recommendations in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

References

[1] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *SIGKDD*. ACM, 2003, pp. 137–146.
[2] W. Chen, L. V. Lakshmanan, and C. Castillo, "Information and influence propagation in social networks," *Synthesis Lectures on Data Management*, vol. 5, no. 4, pp. 1–177, 2013.
[3] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of computer computations*. Springer, 1972, pp. 85–103.
[4] W. Chen, C. Wang, and Y. Wang, "Scalable influence maximization for prevalent viral marketing in large-scale social networks," in *SIGKDD*. ACM, 2010, pp. 1029–1038.
[5] W. Chen, Y. Yuan, and L. Zhang, "Scalable influence maximization in social networks under the linear threshold model," in *ICDM*. IEEE, 2010, pp. 88–97.
[6] S. Khot, "Ruling out ptas for graph min-bisection, dense k-subgraph, and bipartite clique," *SIAM Journal on Computing*, vol. 36, no. 4, pp. 1025–1071, 2006.
[7] V. V. Vazirani, *Approximation algorithms*. Springer Science & Business Media, 2013.
[8] A. Bhaskara, M. Charikar, E. Chlamtac, U. Feige, and A. Vijayaraghavan, "Detecting high log-densities: an o (n $1/4$) approximation for densest k-subgraph," in *STOC*. ACM, 2010, pp. 201–210.
[9] W. Lu, W. Chen, and L. V. Lakshmanan, "From competition to complementarity: comparative influence diffusion and maximization," *VLDB*, vol. 9, no. 2, pp. 60–71, 2015.
[10] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functionsi," *Mathematical Programming*, vol. 14, no. 1, pp. 265–294, 1978.
[11] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier, "Maximizing social influence in nearly optimal time," in *SODA*. Society for Industrial and Applied Mathematics, 2014, pp. 946–957.
[12] Y. Tang, Y. Shi, and X. Xiao, "Influence maximization in near-linear time: A martingale approach," in *SIGMOD*. ACM, 2015, pp. 1539–1554.
[13] Y. Tang, X. Xiao, and Y. Shi, "Influence maximization: Near-optimal time complexity meets practical efficiency," in *SIGMOD*. ACM, 2014, pp. 75–86.
[14] H. T. Nguyen, T. N. Dinh, and M. T. Thai, "Cost-aware targeted viral marketing in billion-scale networks," in *INFOCOM*. IEEE, 2016.
[15] P. Dagum, R. Karp, M. Luby, and S. Ross, "An optimal algorithm for monte carlo estimation," *SIAM Journal on computing*, vol. 29, no. 5, pp. 1484–1496, 2000.
[16] M. Mitzenmacher and E. Upfal, *Probability and computing: Randomized algorithms and probabilistic analysis*. Cambridge University Press, 2005.
[17] H. T. Nguyen, M. T. Thai, and T. N. Dinh, "Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks," in *SIGMOD*. ACM, 2016.
[18] P. Domingos and M. Richardson, "Mining the network value of customers," in *SIGKDD*. ACM, 2001, pp. 57–66.
[19] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance, "Cost-effective outbreak detection in networks," in *SIGKDD*. ACM, 2007, pp. 420–429.
[20] N. Du, L. Song, M. Gomez-Rodriguez, and H. Zha, "Scalable influence estimation in continuous-time diffusion networks," in *NIPS*, 2013, pp. 3147–3155.
[21] E. Cohen, D. Delling, T. Pajor, and R. F. Werneck, "Sketch-based influence maximization and computation: Scaling up with guarantees," in *CIKM*. ACM, 2014, pp. 629–638.
[22] B. Lucier, J. Oren, and Y. Singer, "Influence at scale: Distributed computation of complex contagion in networks," in *SIGKDD*. ACM, 2015, pp. 735–744.
[23] Y. Yang, X. Mao, J. Pei, and X. He, "Continuous influence maximization: What discounts should we offer to social network users?" in *SIGMOD*. ACM, 2016.

TABLE 4
Influence spread and information activity on the DBLP and LiveJournal data sets

Data	IC model						LT model					
	k=20			k=200			k=20			k=200		
	influence	activity	ratio	influence	activity	ratio	influence	activity	ratio	influence	activity	ratio
DBLP	2,291	3,509	1.53	13,764	21,165	1.54	2,834	5,179	1.83	17,445	32,712	1.88
LiveJ	66,615	958,95	1.44	186,726	333,598	1.79	89,559	184,842	2.06	297,014	839,334	2.83

[24] S. Chen, J. Fan, G. Li, J. Feng, K.-l. Tan, and J. Tang, "Online topic-aware influence maximization," *VLDB*, vol. 8, no. 6, pp. 666–677, 2015.

[25] Q. Liu, B. Xiang, N. J. Yuan, E. Chen, H. Xiong, Y. Zheng, and Y. Yang, "An influence propagation view of pagerank," *ACM Trans. on KDD*, vol. 11, no. 3, p. 30, 2017.

[26] A. Goyal, W. Lu, and L. V. Lakshmanan, "Simpath: An efficient algorithm for influence maximization under the linear threshold model," in *ICDM*. IEEE, 2011, pp. 211–220.

[27] K. Jung, W. Heo, and W. Chen, "Irie: Scalable and robust influence maximization in social networks," in *ICDM*.

[28] Q. Liu, B. Xiang, E. Chen, H. Xiong, F. Tang, and J. X. Yu, "Influence maximization over large-scale social networks: A bounded linear approach," in *CIKM*. ACM, 2014, pp. 171–180.

[29] S. Cheng, H. Shen, J. Huang, W. Chen, and X. Cheng, "Imrank: influence maximization via finding self-consistent ranking," in *SIGIR*. ACM, 2014, pp. 475–484.

[30] A. Goyal, F. Bonchi, and L. V. Lakshmanan, "A data-based approach to social influence maximization," *VLDB*, vol. 5, no. 1, pp. 73–84, 2011.

[31] W. Chen, W. Lu, and N. Zhang, "Time-critical influence maximization in social networks with time-delayed diffusion process," in *AAAI*. AAAI Press, 2012, pp. 592–598.

[32] Y. Sun, R. Zhang, A. Y. Xue, J. Qi, and X. Du, "Reverse nearest neighbor heat maps: A tool for influence exploration," in *ICDE*. IEEE, 2016, pp. 966–977.

[33] J. Huang, Z. Wen, J. Qi, R. Zhang, J. Chen, and Z. He, "Top-k most influential locations selection," in *CIKM*. ACM, 2011, pp. 2377–2380.

[34] T. Xia, D. Zhang, E. Kanoulas, and Y. Du, "On computing top-t most influential spatial sites," in *VLDB*. VLDB Endowment, 2005, pp. 946–957.

[35] F. Tang, Q. Liu, H. Zhu, E. Chen, and F. Zhu, "Diversified social influence maximization," in *ASONAM*. IEEE, 2014, pp. 455–459.

[36] S. Bhagat, A. Goyal, and L. V. Lakshmanan, "Maximizing product adoption in social networks," in *WSDM*. ACM, 2012, pp. 603–612.

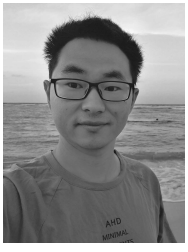
[37] Z. Wang, E. Chen, Q. Liu, Y. Yang, Y. Ge, and B. Chang, "Maximizing the coverage of information propagation in social networks," in *IJCAI*. AAAI Press, 2015, pp. 2104–2110.

[38] T. Xu, D. Liu, E. Chen, H. Cao, and J. Tian, "Towards annotating media contents through social diffusion analysis," in *ICDM*. IEEE, 2012, pp. 1158–1163.

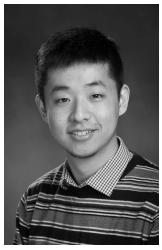
[39] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "Arnetminer: extraction and mining of academic social networks," in *SIGKDD*. ACM, 2008, pp. 990–998.

[40] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," *Knowledge and Information Systems*, vol. 42, no. 1, pp. 181–213, 2015.

[41] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: bringing order to the web." 1999.



Zhefeng Wang is currently a Ph.D. Candidate at the School of Computer Science and Technology, University of Science and Technology of China. He received a B.E. degree from University of Science and Technology of China, China in 2012. His research interests include social network, social media analysis, machine learning, and text mining. He has published several papers in refereed conference proceedings and journals such as SIGIR, KDD, IJCAI and TKDE.



Yu Yang received his B.E. degree from Hefei University of Technology in 2010, and his M.E. degree from University of Science and Technology of China in 2013, both in Computer Science. He is currently a Ph.D. student in School of Computing Science at Simon Fraser University, Canada. His research interests lie in algorithmic aspects of data mining, with an emphasis on managing and mining dynamics of large scale networks.

Jian Pei (Fellow) is a Professor at the School of Computing Science at Simon Fraser University, Canada. His research interests can be summarized as developing effective and efficient data analysis techniques for novel data intensive applications. He is currently interested in various techniques of data mining, Web search, information retrieval, data warehousing, online analytical processing, and database systems, as well as their applications in social networks, health-informatics, business and bioinformatics. His research has been supported in part by government funding agencies and industry partners. He has published prolifically and served regularly for the leading academic journals and conferences in his fields. He is a Fellow of the Association for Computing Machinery (ACM) and the Institute of Electrical and Electronics Engineers (IEEE). He is the recipient of several prestigious awards.

Lingyang Chu is currently a Postdoctoral Fellow at the School of Computing Science at Simon Fraser University, Canada. He received his doctoral degree from the University of Chinese Academy of Sciences, Beijing, China in 2015. His research interests include large scale network analysis, multimedia information retrieval and cross modality information processing. He has published several papers in refereed conference proceedings and journals, such as KDD, VLDB, TMM and TCSVT.

Enhong Chen (Senior Member) received the PhD degree from the University of Science and Technology of China (USTC). He is a professor and vice dean of the School of Computer Science and Technology at USTC. His general area of research includes data mining, personalized recommendation systems, and web information processing. He has published more than 150 papers in refereed conferences and journals. His research is supported by the National Natural Science Foundation of China, National High Technology Research and Development Program 863 of China, etc. He is a program committee member of more than 40 international conferences and workshops. He is a senior member of the Institute of Electrical and Electronics Engineers (IEEE).