A Second-Order Diffusion Model For Influence Maximization In Social Networks

Wenyi Tang, Guangchun Luo, *Member, IEEE*, Yubao Wu, Ling Tian, Xu Zheng, and Zhipeng Cai, *Senior Member, IEEE*

***Abstract*—In social networks, several influential individuals can promote an idea or a product to numerous individuals. Thus, it is valuable to solve the Influence Maximization (IM) problem, which asks for finding the most influential set of individuals in a social network. To estimate the influence of individuals, the existing Independent Cascade (IC) model simulates the influence diffusion only considering the influences from direct in-neighbors to nodes. This consideration does not hold in the real life. In many cases, people are likely influenced by an information depending on where it comes from, instead of who gives it. To simulate the influence diffusion more accurate, this paper proposes the second-order IC model, which takes the previous influence into consideration. In addition, we design an approximate algorithm and its distributed extension for IM under the second-order IC model. Experimental results show that our second-order IC model outperforms the IC model in terms of simulating influence diffusions. The proposed algorithms are efficient, and obtained node sets are influential.**

***Index Terms*—influence maximization, diffusion model, second order**

I. Introduction

In social networks, information diffusions of some products or ideas can result in adoptions of products or ideas through the word of mouth effect [1]–[7]. Since we cannot directly propagate the information to everyone in a network, influential individuals are usually selected to propagate the information [8], [9]. Therefore, IM problem is formulated, which asks for finding a fixed number of most influential individuals in a network [10]. To estimate the influence, Kempe *et al.* [10] propose two diffusion models, IC model and Linear Threshold (LT) model. But these diffusion models only consider impacts from nodes' direct in-neighbors, losing the information of previous activations. Therefore, this paper focuses on second-order influence diffusions, which take previous activations into consideration.

This work is partially supported by the Sichuan Science and Technology Program (No.2017JZ0031, 2018JY0073, 2018HH0075, 2016FZ0108, 2018JY0067), by the Chengdu Science and Technology Bureau Program (No.2018-YF09-00051-SN), by the Young Scientists Fund of the National Natural Science Foundation of China (Grant No. 61802050). (Corresponding author: Guangchun Luo.)

W. Tang, L. Tian and X. Zheng are with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: yigetangwenyi@outlook.com; {lingtian, xzheng}@uestc.edu.cn).

G. Luo is with the School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: gcluo.uestc@gmail.com).

Y. Wu and Z. Cai are with the Department of Computer Science, Georgia State University, Atlanta 30303, USA (e-mail: {ywu28, zcai}@gsu.edu).

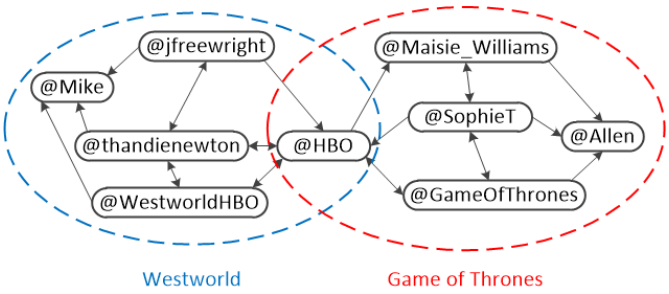


Fig. 1. An example of the Twitter follower network

TABLE I
Retweet Number of @ShopieT's Posted 167 Tweets

	@GameOfThrown	@WestworldHBO
the real Twitter network	11	0
the prediction of IC	7.6	6.2
the prediction of 2nd IC	7.2	0

Under the IC and the LT model, Kempe *et al.* [10] prove that IM is a NP-Hard problem, and design a greedy algorithm which returns an $(1 - \frac{1}{e} - \epsilon)$-approximate solution. For the higher efficiency, other researchers design numerous kinds of algorithms for IM, such as greedy algorithms [11], [12], heuristic algorithms [13]–[19], and Reverse Influence Sampling (RIS) based algorithms [20]–[22].

As a widely used diffusion model, the IC mdoel assumes that the activation of any node only depends on impacts from its direct in-neighbors, independent of impacts from previous activations. In fact, this assumption of the IC model considerably differs from the reality, while our proposed second-order IC model is close to the reality. It is because that the second-order IC model considers impacts from previous activations, which contains more information of influence relationships. For example, Fig. 1 shows a subgraph of the Twitter follower network. Each node in the graph denotes a user, and each directed edge denotes a following relation, for instance, user @Mike follows user @WestworldHBO. There are two communities about the TV shows, The Westworld community on the left and the Game of Thrones (GoT) community on the right. If an actress in GoT (@SophieT) posts a tweet, it is likely to be propagated in the GoT community instead of the Westworld community. According to the tweet cascade data in real-life, @SophieT posted 167 tweets in 2017. There are 11 @SophieT's tweets propagated from @HBO to @GameOfThrones, and 0 @SophieT's tweets propagated from @HBO to @WestworldHBO. However, given that @SophieT

posted 167 tweets, IC model predicts 7.6 tweet cascades from @HBO to @GameOfThrones and 6.2 tweet cascades from @HBO to @WestworldHBO, respectively. Ttweet cascades reflect influence diffusions in the Twitter. Prediction results of the IC indicates that @SophieT influences two different communities similarly, which considerably differs from the reality. Given that @SophieT posted some tweets, second-order IC model predicts that @SophieT influences GoT community much more than Westworld community, as shown in Table I, which is closer to the reality than the IC model.

As far as we know, existing diffusion models only consider impacts from nodes' direct in-neighbors, losing the information of previous activations, which may lead to the inaccurate simulation of real influence diffusions. Therefore, this paper aims at designing a new diffusion model, taking previous activations into consideration, which improves the simulation accuracy of real influence diffusions. However, how to design this kind of diffusion model is challenging. Based on that influence diffusions in the traditional IC model are activations from node to node, we design the second-order influence diffusions as activations from edge to edge. Combined with some other rules of influence propagations, we design the second-order IC model, which considerably improves the simulation accuracy of real influence diffusions.

In addition, we design an RIS based algorithm, called IMM_2nd, for IM under the second-order IC model. Since the second-order IC model takes previous activations into consideration, it increases the complexity of solving IM problem compared with the IC model. In order to reduce the practical running time, we design a distributed extension of IMM_2nd algorithm which achieves comparable efficiency with algorithms based on the IC model. Furthermore, the obtained seed sets by both of our proposed algorithms have comparable influence with algorithms based on IC model.

The main contributions of this paper are as follows:

1) To the best of our knowledge, it is the first time to design a second-order diffusion model which takes previously activated nodes into consideration.
2) According to a theoretical study, we define the second-order influence probability. Then, we design the second-order IC model based on the traditional IC model. Moreover, we show the relation between the second-order IC model and IC model in terms of our defined augmented-line graph.
3) We prove that IM under the second-order IC is a NP-Hard problem. Then, we design IMM_2nd which is an approximate algorithm based on the RIS method. In order to achieve a higher efficiency, we propose the DIMM_2nd algorithm as a distributed extension of the IMM_2nd. Proposed algorithms are efficient, and guarantee the $(1-\frac{1}{e}-\epsilon)$-approximation ratio with a high probability.
4) Extensive experiments on real networks show that the developed second-order IC model simulates influence diffusions in real networks more accurate than IC model. The developed IMM_2nd and DIMM_2nd algorithms are efficient.

II. Related Work

Kempe *et al*. firstly define the IC model and LT model to estimate influence, and design a greedy algorithm for IM based on a Monte Carlo approach [10]. It guarantees to return an $(1-\frac{1}{e}-\epsilon)$-approximate solution and holds $O(knm\pi)$ time complexity. Note that ϵ is the error generated by estimating the influence of nodes, and e is the base of the natural logarithm.

Based on IC model and LT model, numerous algorithms are proposed to improve the efficiency for IM [11]–[28]. Greedy algorithms [11], [12] follow the same greedy framework with Kempe *et al*.'s algorithm [10]. They reduce the running time by using branch and bound approaches, which omit the influence estimations of some uninfluential node sets. Their solutions retain the same $(1-\frac{1}{e}-\epsilon)$-approximation ratio, while these greedy algorithms still hold $O(knm\pi)$ theoretical complexity, and cannot scale to large networks.

Heuristic algorithms [13]–[19], [29] take less running time in practice, while none of them retains the $(1-\frac{1}{e}-\epsilon)$-approximation ratio. For the IC model, Chen *et al*. [29] increase the efficiency based on an influence estimation. For the LT model, Chen *et al*. [14] approximate the influence regions of nodes by using local directed acyclic graphs. Another algorithms [20]–[22] estimate the influence by RIS method, instead of the traditional Monte Carlo approach. This significantly improves the efficiency and retains the $(1-\frac{1}{e}-\epsilon)$-approximation ratio at the same time.

In addition, some researchers extend the traditional IC model or LT model. Zhu *et al*. [30] consider the product price in viral marketing, and extend the IC model and LT model to a price related frame. Nicola *et al*. [31] propose a topic-aware extension of IC model, in which the influence probability depends on different topics instead of being constant. Based on LT model, the Opinion-based Cascading (OC) model in [32] considers both positive and negative influences. Zhou *et al*. [33] present the Two Phase (TP) model, a location-based extension of IC model. TP model considers both online influence and offline influence. It accurately describes the process of accepting products under the scenario, where products are promoted online while customers purchase in offline shops.

The existing diffusion models only consider the impacts from nodes' direct in-neighbors, and there is no research about second-order diffusion models. To simulate real influence diffusions more accurate, this paper considers the impacts from previous activations, and extents the IC model based on the second-order influence diffusions. The most related work of ours is that, in the field of random walk, Wu *et al*. [34] propose the second-order proximity measure which takes the previously visited node into consideration.

III. Preliminaries

In this section, we introduce the IC model [10], and show the formal definition of IM problem. Finally, we present an overview of the Kempe *et al*.'s greedy framework [10] and the RIS framework [20]. The main symbols used in this paper and their definitions are listed in Table II.

TABLE II
MAIN SYMBOLS

symbols	definitions
$G(V,E,P)$	directed graph G with node set V, edge set E, and probability set P
n,m,σ	number of nodes; number of edges; $\sigma=\sum_{i\in V}\|I_i\|\cdot\|O_i\|$
I_i , O_i	set of in-/out-neighbors of node i
$O_{(i,j)}$	set of out-edges of edge (i,j)
d_i	the indegree of node i
σ_i	the number of path of length two ending at node i
$I(S)$	the influence of a node set S in a diffusion process
$\mathbb{E}[\cdot]$	the expectation of a random variable
$p_{i,j}$	influence probability from node i to j
$p_{i,j,k}$	influence prob. from j to k if influence came from i
$p_{u,v}$	influence prob. from edge u to v, $p_{(i,j),(j,k)}=p_{i,j,k}$
R_j	the RR set
$\mathcal{R}$	a collection of sampled RR sets

Algorithm 1 Simulation_1stIC(G,S)

Input: graph $G(V,E,P)$, seed set (S)
Output: set of active nodes (A)

```
A ← ∅, B_t ← S, C_{t+1} ← ∅;
while B_t ≠ ∅ do
    for each active node i ∈ B_t do
        for each inactive node j ∈ O_i do
            if p_{i,j} ≥ c, random number c ∈ [0,1] then
                j becomes active, C_{t+1} ← C_{t+1} ∪ {j};
    A ← A ∪ B_t, B_t ← C_{t+1}, C_{t+1} ← ∅;
return A;
```

A. Independent Cascade Model

In general, a social network is abstracted as a weighted graph $G=(V,E,P)$ with $|V|=n$ nodes and $|E|=m$ directed edges. For any edge (i,j), i is the in-neighbor of j, and j is the out-neighbor of i. Each edge $(i,j)\in E$ is associated with a weight $p_{i,j}\in P$ and $p_{i,j}\in[0,1]$, which indicates the probability that node i successfully activates node j.

As a widely used diffusion model, the IC model [10] assumes that every node is inactive at first. Given an initial set of active nodes, called seed nodes, the propagation of influence starts from seed nodes to the whole graph. Three basic rules of the influence propagation are as follows:

1) **non-reverse activation**: once a node becomes active, it never becomes inactive;
2) **single chance**: once a node becomes active at timestamp t, it only has a single chance to activate its inactive out-neighbors at timestamp $t+1$;
3) **constant probability**: the probability $p_{i,j}$ is constant, independent with the timestamp.

The simulation of an IC process is shown in Algorithm 1. At each timestamp t, the newly activated nodes has a single chance to activate their inactive out-neighbors. Influence diffusions end until there is no activation in the network. Since the worst case is that each node and each edge is traversed once, Algorithm 1 takes $O(m)$ time.

B. Problem Definition

Given a set S_k of k seed nodes, the number of all active nodes at the end of a diffusion process is denoted as $I(S_k)$. We say $\mathbb{E}[I(S_k)]$ is the influence of the seed set S_k. IM problem is defined as following,

Definition 1 (Influence Maximization [10]): Given a graph $G=(V,E,P)$, $k\in Z^+$ and a diffusion model, the Influence Maximization problem asks for a seed set $S_k\in V$ of k seed nodes that maximizes its influence $\mathbb{E}[I(S_k)]$.

C. Kempe et al.'s Greedy Framework

Under the IC model, the IM is a NP-Hard problem [10]. Since $\mathbb{E}[I(\cdot)]$ can be denoted as a monotone submodular function [10], IM problem is essentially a problem of maximizing a monotone submodular function.

Kempe *et al.* use the standard greedy strategy [35] of maximizing a monotone submodular function to solve the IM problem. It iteratively adds a node with the maximum incremental influence into the seed set, until the size of the seed set achieves k. To estimate the influence, Kempe *et al.* use a Monte Carlo method. It repeats the simulation of the IC process π times, and uses the sample mean as an estimator. According to the law of large numbers, for any $\epsilon>0$,

$$\lim_{\pi\to\infty}\mathbb{P}[|\frac{1}{\pi}\sum_{i=1}^{\pi}I(S_k)-\frac{1}{\pi}\sum_{i=1}^{\pi}\mathbb{E}[I(S_k)]|<\epsilon]=1. \quad (1)$$

When the simulation time π tends to be infinite, the sample mean tends to the influence $\mathbb{E}[I(S_k)]$.

The Kempe *et al.*'s greedy algorithm takes $O(k\pi nm)$ time in total, and it guarantees an $(1-\frac{1}{e}-\epsilon)$-approximate solution, where ϵ is the error generated by the Monte Carlo simulations. Subsequently, some researchers derive more efficient algorithms following the Kempe *et al.*'s greedy framework [11], [12]. However, these improved algorithms still take considerable running time, due to the significant computation overheads of the influence estimation. Specifically, when they select the first seed node, they must estimate the influence of each node $i\in V$. This takes considerable $O(\pi nm)$ time.

D. Reverse Influence Sampling Framework

In order to overcome the inefficiency of Kempe *et al.*'s greedy framework, Borgs *et al.* [20] introduce the RIS, a novel sampling method for the IM. It captures the influence landscape of a graph by generating Reverse Reachable (RR) sets.

Definition 2 (Reverse Reachable Set [36]): Given $G=(V,E,P)$, an RR set R_j is generated from G by: 1) randomly selecting a node $j\in V$ as the destination; 2) generating a new graph g as same as G; 3) deleting each edge (i,j) in g with probability $1-p_{i,j}$ and 4) returning R_j as the set of nodes that can reach j in g.

Fig. 2 shows an example of generating an RR set. For R_1, node l is the randomly chosen destination. Edge (j,k) is deleted with probability $1-p_{j,k}=0.5$. Thus, $R_1=\{i,k,l\}$ contains all nodes which can reach l. R_2 is generated in the same way.

Fig. 2. An example of generating RR sets under IC model

Algorithm 2 RIS method

Input: graph $G(V,E,P)$, number of seed node (k), threshold (θ)
Output: set of seed nodes (S_k)

$S_k \leftarrow \emptyset$, $\mathcal{R} \leftarrow \emptyset$, generate G^T by reversing each edge in G;
while $|\mathcal{R}| < \theta$ **do**
  randomly choose a destination $j \in V$;
  $\mathcal{R} \leftarrow \mathcal{R} \cup$ Simulation_1stIC($G^T, \{j\}$);
while $|S_k| < k$ **do**
  $j \leftarrow \arg\max_{i \in V} Cover_{\mathcal{R}}(\{i\})$, $S_k \leftarrow S_k \cup \{j\}$;
  delete the RR sets in $\mathcal{R}$ which contain j;
return S_k;

Consider that each edge in G is reversed to generate a reverse graph G^T. Given a diffusion process of IC model, the nodes that can activate the destination in G are the nodes activated by the destination in G^T.

Borgs *et al*. [20] use the coverage of RR sets as the estimator of the influence. Given a collection of RR sets $\mathcal{R}$, IM problem is equivalent to a node selection problem which asks for a seed set S_k maximizing the coverage of $\mathcal{R}$. Since $Cover_{\mathcal{R}}(S_k)$ is a monotone submodular function [20], the RIS based algorithms [20]–[22] alternatively resolve the IM problem by following two steps,

1) **Sampling**: generate a collection of RR sets from G.
2) **Node Selection**: use the standard greedy strategy of maximizing a monotone submodular function to obtain a seed set S_k.

This two-step algorithm is shown in Algorithm 2, where $Cover_{\mathcal{R}}(S)$ denotes the number of RR sets $R_j \in \mathcal{R}$ intersecting with S, i.e., the coverage of $\mathcal{R}$. Different RIS based algorithms [20]–[22] all guarantee an $(1-\frac{1}{e}-\epsilon)$-approximation ratio, and their complexity depend on the size of $\mathcal{R}$.

IV. Second-Order Influence Diffusion

In this section, we firstly define the second-order influence probability. Then we propose the second-order IC model based on this probability. Finally, we show the relation between the IC model and the second-order IC model.

A. Second-Order Influence Probability

In first-order IC model, influence propagates from an active node i to an inactive node j with probability $p_{i,j}$. Let $\mathbb{X}_i^t$ be a random variable that equals to 1 if node i is activated at timestamp t, and equals to 0 if node i is inactive at timestamp t. The first-order influence probability can be represented as a conditional probability,

$$p_{i,j} = \mathbb{P}[\mathbb{X}_j^t = 1, i \to j | \mathbb{X}_i^{t-1} = 1], \tag{2}$$

where $i \to j$ indicates node j is activated by node j.

In second-order IC model, we need to consider the source of the influence, i.e., the previous activated node which activates current active node. We use $p_{i,j,k}$ to represent the second-order influence probability from node j to k, given that the previous activation was from node i to j,

$$\begin{aligned} p_{i,j,k} &= \mathbb{P}[\mathbb{X}_k^{t+1} = 1, j \to k, i \to j | \mathbb{X}_i^{t-1} = 1, \mathbb{X}_j^t = 1] \\ &= \mathbb{P}[\mathbb{X}_j^t = 1, \mathbb{X}_k^{t+1} = 1, j \to k | \mathbb{X}_i^{t-1} = 1, \mathbb{X}_j^t = 1, i \to j]. \end{aligned} \tag{3}$$

Let $\mathbb{Y}_{(i,j)}^t = 1$ represent the joint event $(\mathbb{X}_i^{t-1} = 1, \mathbb{X}_j^t = 1, i \to j)$, which means node j is activated by node i at timestamp t. We define edge $u = (i,j)$ is activated at timestamp t, when joint event $(\mathbb{X}_i^{t-1} = 1, \mathbb{X}_j^t = 1, i \to j)$ happens. Based on the concept of edge activation, the second-order influence probability is,

$$p_{i,j,k} = \mathbb{P}[\mathbb{Y}_{(j,k)}^{t+1} = 1 | \mathbb{Y}_{(i,j)}^t = 1] = p_{u,v}, \tag{4}$$

where u is the edge (i,j), and v is the edge (j,k). Therefore, $p_{i,j,k}$ describes the activation in an edge-to-edge perspective, which is the foundation of the second-order IC model.

B. Second-Order IC Model

For arbitrary edges (i,j) and (l,k), if node j and node l are the same node, (l,k) is the out-edge of (i,j), and (i,j) is the in-edge of (l,k). When arbitrary active node i activates node j successfully, both node j and edge (i,j) become active. Namely, an active edge (i,j) means that both node i and j are active.

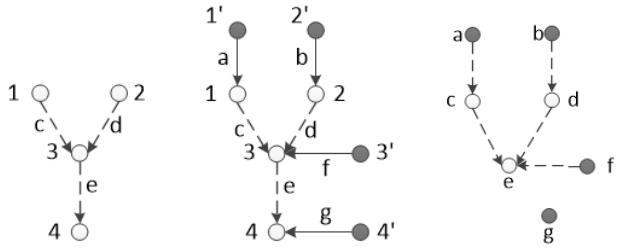
In second-oder IC model, every node and edge are inactive at first. Given a seed set, seed nodes activate each of their out-neighbors (both active and inactive) with first-order influence probability (node-to-node activation). The activations of seed nodes make some edges become active. After that, influence propagates from edge to edge, i.e., the active edge $u = (i,j)$ activates its inactive out-edges $v = (j,k)$ with probability $p_{u,v} = p_{i,j,k}$. If the activation succeeds, both node k and edge $v = (j,k)$ become active.

The basic rules of influence propagation in second-order IC model are as follows:

1) **non-reverse activation**: once a node or edge becomes active, it never becomes inactive;
2) **single chance**: a seed node only has a single chance to activate each of its out-neighbors (node-to-node); after the activations of seed nodes, once an edge becomes active at timestamp t, it only has a single chance to activate its inactive out-edges at timestamp $t+1$ (edge-to-edge);
3) **constant probability**: the probabilities $p_{s,j}$ and $p_{u,v} = p_{i,j,k}$ are constant, independent with the timestamp.

These three basic rules of the second-order IC are similar to the basic rules of the IC. Each rule extends the content about the edge-to-edge activation. At first, we add the non-reverse activation rule of any edge. Secondly, we add the single chance rule of any edge-to-edge activation. Finally, we add the constant probability rule of second-order probability.

Compared with the single chance rule in the IC, activations of seed nodes and non-seed nodes are different in the single chance rule of the second-order IC.



(a) Original graph (b) Intermediate step (c) Augmented-line graph

Fig. 3. Original graph and augmented-line graph

Algorithm 3 Simulation_2ndIC_1(G, S)

Input: graph $G(V,E,P)$, set of seed nodes (S)
Output: set of active nodes (A)

1: $A \leftarrow \emptyset$, $B \leftarrow S$, $C_t \leftarrow \emptyset$, $D_{t+1} \leftarrow \emptyset$;
2: **while** $B \neq \emptyset$ **do**
3: choose an arbitrary node $i \in B$, $A \leftarrow A \cup \{i\}$;
4: **for** each node $j \in O_i$ **do**
5: **if** $p_{i,j} \geq c$, random number $c \in [0,1]$ **then**
6: (i,j) is active, $C_t \leftarrow C_t \cup \{(i,j)\}$;
7: $B \leftarrow B - \{i\}$;
8: **while** $C_t \neq \emptyset$ **do**
9: **for** each active edge $(i,j) \in C_t$ **do**
10: $A \leftarrow A \cup \{j\}$;
11: **for** each inactive edge $(j,k) \in O_{(i,j)}$ **do**
12: **if** $p_{i,j,k} \geq c$, random number $c \in [0,1]$ **then**
13: edge (j,k) is active, $D_{t+1} \leftarrow D_{t+1} \cup \{(j,k)\}$;
14: $C_t \leftarrow D_{t+1}$, $D_{t+1} \leftarrow \emptyset$;
15: **return** A;

For a seed node, it activates each of its out-neighbors (whether active or not), with first-order probability. For example in Fig. 3(a), assume that seed node 1 activates node 3 at first, and it succeeds. Edge c becomes active, namely node 3 becomes active. When seed node 2 starts activation, it still activates the active node 3. If it succeeds, edge d will become active. This is different with the corresponding rule of IC. In above example, when seed node 1 has already activated node 3 successfully, seed node 2 will not activate the active node 3 under IC.

For a non-seed node, influence propagates under second-order IC model from edge to edge. Combined with the example in Fig. 3(a), assume that both seed node 1 and 2 active node 3 successfully. Non-seed node 3 has two active in-edges $c = (1,3)$ and $d = (2,3)$. Active edge $c = (1,3)$ activates inactive edge $d = (3,4)$ first with second-order probability $p_{c,e} = p_{1,3,4}$. If the activation succeeds, edge e becomes active. When d executes its single chance of activation, it will not activate the already active edge e. Since e has no out-edge, there is no activation in this network. The propagation of influence ends in this graph.

Therefore, under second-order IC model, an active node may have multiple active in-edges and out-edges. An active edge may have multiple active in-edges, but it is only activated by one of them.

Algorithm 3 shows how to simulate the second-order IC process. Set B contains all seed nodes. Set C_t contains newly activated edges at timestamp t, which have a chance to activate their out-edges at timestamp $t+1$. Set D_{t+1} contains newly activated edges at timestamp $t+1$. Set A contains all active nodes. $O_{(i,j)}$ denotes the out-edge set of edge (i,j). At timestamp 0 (line 1), B contains all seed nodes. Set A, C_t, D_{t+1} are empty. At timestamp 0, any seed node i in B (line $2 \sim 3$) has a single chance to activate each of its out-neighbors j (line $4 \sim 5$), succeeding with first-order influence probability $p_{i,j}$. If the activation succeeds, node j becomes or remains active, and (i,j) becomes newly active edge in C_t (line 6). After timestamp 0, each edge (i,j) in C_t tries to activate its inactive out-edges (j,k), succeeding with second-order influence probability $p_{i,j,k}$ (line $8 \sim 11$). If the activation succeeds, edge (j,k) becomes a newly active edge in D_{t+1}. Influence propagates in this way, until there is no newly active edge.

Line $2 \sim 6$ takes $O(m)$ time. The worst case of line $8 \sim 12$ is that every path of length two in the graph executes an activation. This takes $O(\sigma)$ time, where σ is the number of paths of length two in G. Thus, Algorithm 3 takes $O(\sigma)$ time in total.

C. Relation to First-Order IC

In order to explain the relation between the IC and the second-order IC, we define the augmented-line graph, which is slightly different with the traditional line graph.

Definition 3: [Augmented-Line Graph] Given a graph $G(V,E,P)$, its augmented-line graph $G'(V',E',P')$ is constructed as follows,

1) for each node $i \in V$, add an in-neighbor i' and an in-edge (i',i), for each $j \in O_i$, set $p_{i',i,j} = p_{i,j}$;
2) add each edge $u = (i,j) \in E$ as a node into V', each path of length two in G as an directed edge into E', and each $p_{u,v} = p_{i,j,k}$ into P';

Note that, if node i in G is a seed node, the corresponding node i' and edge (i',i) is initially active, and the node $a = (i',i)$ in G' is the seed node. We call node i' is the augmented leaf of node i, and edge (i',i) is the augmented edge of node i.

For example, Fig. 3(a) shows an original graph. We add in-neighbors $1', 2', 3', 4'$ and in-edges $a = (1',1), b = (2',2), f = (3',3), g = (4',4)$, and set $p_{1',1,3} = p_{1,3}, p_{2',2,3} = p_{2,3}, p_{3',3,4} = p_{3,4}$, as shown in Fig. 3(b). Then edges in Fig. 3(b) become the nodes, and paths of length two become the directed edges in augmented-line graph (Fig. 3(c)). If node 1 is a seed node in Fig. 3(a), then node $1'$ and edge $(1',1)$ is initially active in Fig. 3(b), and node a is a seed node in Fig. 3(c).

After the above definition, we present the relation between second-order IC and IC,

Theorem 1: The second-order IC process in the original graph is equivalent to the IC process in corresponding augmented-line graph.

Proof: Let G_a be the leaf-augmented graph of original graph G, obtained by adding an in-neighbor i' and an in-edge (i',i) for each node i in G. Set each $p_{i',i,j} = p_{i,j}$ in G_a. For example, Fig. 3(b) shows the leaf-augmented graph of Fig. 3(a). If node i is a seed node, augmented node i' and edge (i',i) are active. Consider a path $i \rightarrow j \rightarrow k$ in G, it corresponds

Algorithm 4 Simulation_2ndIC_2(G, S)

Input: graph $G(V, E, P)$, set of seed nodes (S)
Output: set of active nodes (A)

1: generate augmented-line graph $G'(V', E', P')$ from G, $S' \leftarrow \emptyset$;
2: **for** each node $s \in S$ **do** $(s', s) = u \in V'$ is active, $S' \leftarrow S' \cup \{u\}$;
3: $B \leftarrow$ Simulation_1stIC(G', S');
4: **for** each $u \in B$ corresponding to $(i, j) \in E$ **do** $A \leftarrow A \cup \{i, j\}$;
5: **return** A;

to the path $i' \rightarrow i \rightarrow j \rightarrow k$ in G_a. When i is a seed node in G, i activates j with $p_{i,j}$. If it succeeds, activated edge (i, j) activates edge (j, k) with $p_{i,j,k}$. This equals to the process in G_a that initially active edge (i', i) successfully activates edge (i, j) with $p_{i',i,j} = p_{i,j}$, then activated edge (i, j) tries to activate edge (j, k) with $p_{i,j,k}$. Such process starting from initially active edges is the second-order IC process in G_a, which is equivalent to the second-order IC process in G.

We consider the second-order IC process in G_a. For any edge (j, k), let A denote the event that (j, k) is inactive at timestamp t, and B denote the event that (j, k) is active at timestamp $t + 1$. Event A means that edge (j, k) may be inactive after timestamp t. So the probability $p'_{(j,k)}$ that edge (j, k) is activated at timestamp $t+1$ is

$$p'_{(j,k)} = \mathbb{P}[B|A] = \frac{\mathbb{P}[A, B]}{\mathbb{P}[A]}. \tag{5}$$

Joint event (A, B) means that the in-edges of (j, k) who are activated before timestamp $t-1$ all fail to activate (j, k), and at least one of (j, k)'s in-edges who are activated at timestamp t successfully activates (j, k) at $t+1$. The probability $p'_{(j,k)}$ is

$$p'_{(j,k)} = \frac{\mathbb{P}[A, B]}{\mathbb{P}[A]} = \frac{1 - \Pi_{(i,j) \in E_t / E_{t-1}} (1 - p_{i,j,k})}{\Pi_{(i,j) \in E_{t-1}} (1 - p_{i,j,k})}, \tag{6}$$

where E_t denotes the set of all active edges at timestamp t.

Similarly, we consider the IC process in corresponding augmented-line graph of G. For any node v, let C denote the event that u is inactive at timestamp t, and D denote the event that u is active at timestamp $t+1$. V_t denotes the set of all active nodes at timestamp t. The probability p'_v that node u is activated at timestamp $t+1$ is

$$p'_v = \mathbb{P}[D|C] = \frac{\mathbb{P}[C, D]}{\mathbb{P}[C]} = \frac{1 - \Pi_{u \in V_t / V_{t-1}} (1 - p_{uv})}{\Pi_{u \in V_t} (1 - p_{uv})}, \tag{7}$$

where V_t denotes the set of all active nodes at timestamp t.

An arbitrary edge $u = (i, j)$ in G_a corresponds to node u in augmented-line graph and $p_{i,j,k} = p_{u,v}$. We obtain,

$$p'_{(j,k)} = p'_v. \tag{8}$$

For any node k in G, the probability that node k becomes active at timestamp $t + 1$ depends on each $p'_{(j,k)} = p'_v$ $(j \in I_k, v = (j, k))$. Thus, for any node k in G, the probability that k becomes active at timestamp $t + 1$ under second-order IC process in G_a equals to the probability that k becomes active at timestamp $t+1$ under IC process in augmented-line graph. Recall that second-order IC process in G_a is equivalent to second-order IC process in G, thus theorem 1 is proved. ■

As a result, there is another way to simulate the second-order IC process, as shown in Algorithm 4. At first, we generate the augmented-line graph, and the corresponding seed set (line $1 \sim 2$). The simulation of IC runs in the augmented-line graph, and set B contains all active nodes in the augmented-line graph (line 3). According to Definition 3, any node $u \in B$ corresponds to an edge $u = (i, j)$ in G. Corresponding node i and j is the active nodes in original graph, and are contained into set A (line 4). Finally, the set A of active nodes in original graph is obtained (line 5). Generation of augmented-line graph takes $O(\sigma)$ time, and the IC process in augmented-line graph takes $O(\sigma)$ time. Therefore, Algorithm 4 takes $O(\sigma)$ time in total, as same as Algorithm 3.

V. Second-order Influence Maximization

In this section, we investigate the IM problem under the developed second-order IC model. At first, we prove that IM under the second-order IC is a NP-Hard problem. Then we design an approximate algorithm based on the RIS method for IM under the second-order IC, and propose a distributed extension of this algorithm.

A. Hardness of IM problem

Theorem 2: IM is a NP-hard problem under the second-order IC model.

Proof: Consider the Set Cover problem. There is a set of all elements $U = \bigcup_{s=1}^{n} u_s$ and a collection of subsets $S_t \in U$, $t = 1, 2, \ldots, m$. Set Cover problem is to decide whether there exist l number of subsets whose union equals to U. An arbitrary instance of Set Cover problem can be constructed as an instance of the IM problem under second-order IC.

We construct a directed graph with $n + 2m$ nodes, as shown in Fig. 4. $2m$ nodes correspond to m subsets, for example, nodes i_1, j_1 correspond to S_1. n nodes correspond to n elements in U, for example, node k_1 corresponds to element u_1. For any node i_t, there is only one edge (i_t, j_t) and $|S_i|$ edges (j_t, k_s), i.e., $S_1 = \{u_1, u_3, u_4\}$, there is an edge (i_1, j_1) and three edges $(j_1, k_1), (j_1, k_3), (j_1, k_4)$ in our construction. For arbitrary edge (i_t, j_t) and arbitrary path of length two $i_t \rightarrow j_t \rightarrow k_s$, we set $p_{i_t,j_t} = 1$ and $p_{i_t,j_t,k_s} = 1$, respectively. The Set Cover problem is equivalent to decide whether there is a seed set S_l of size l in the constructed graph satisfying,

$$\mathbb{E}[I(S_l)] \geq 2l + n. \tag{9}$$

Any seed set S_l of l nodes with influence $2l + n$ uniquely corresponds to a set of subsets whose union is U. Thus, Theorem 2 is proved. ■

B. Generation of RR sets

In order to design an RIS based algorithm for IM problem under second-order IC, we should design the generation of RR sets under second-order IC at first.

Combined with Fig. 3, let node 4 be the destination when we generate an RR set. If node 1 is contained in the RR set, it means that node 1 activates node 3 with $p_{1,3}$ and edge $(1, 3)$

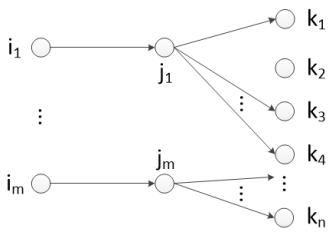


Fig. 4. An example of the construction

Algorithm 5 GenerateRRset_2ndIC(G)

Input: graph $G(V,E,P)$
Output: an RR set (R_j)

1: randomly choose a destination $j \in V$, $R_j \leftarrow \{j\}$, $S \leftarrow \emptyset$;
2: generate augmented-line graph $G'(V',E',P')$ from G;
3: **for** each in-edge $(i,j)=u$ of j **do** $u \in V'$ is active, $S \leftarrow S \cup \{u\}$;
4: generate graph G'^T by reversing each edge in G';
5: $A \leftarrow$Simulation_1stIC(G'^T, S);
6: **for** each $u \in A$ corresponding to an augmented edge (i',i) **do**
7: $\quad R_j \leftarrow R_j \cup \{i\}$;
8: **return** R_j;

activates edge $(3,4)$ with $p_{1,3,4}$ in the diffusion process. Above process corresponds to that we generate the reverse graph G'^T of the augmented-line graph by reversing each edge, and edge is set as the seed node in G'^T. If node a (corresponding to augmented edge $(1',1)$) is activated, this means that e activates c with $p_{c,e}=p_{1,3,4}$ and c activates a with $p_{a,c}=p_{1',1,3}=p_{1,3}$ in the diffusion process.

Based on above description, we design Algorithm 5 to generate an RR set under second-order IC. At first, we randomly choose a destination (line 1), let the RR set contain the destination, and let S be empty. Then we generate augmented-line graph G' from G (line 2). Each node in G' ,corresponding to an in-edge of the destination in G, is set as the seed node. These seed nodes in G' are contained in set S (line 3). The reverse graph G'^T is generated by reversing each edge in G' (line 4). Then we run IC process in G'^T with seed set S (line 5). Set A contains all active nodes in G'^T. Because of the diffusion rules in second-order IC, only the nodes corresponding to augmented edges in A should be contained in the RR set (line $6 \sim 7$).

Under the second-order IC, the RR sets generated by Algorithm 5 also satisfy Lemma 1. Thus, the $n \cdot Cover_{\mathcal{R}}(S)/|\mathcal{R}|$ is an unbiased estimator of $\mathbb{E}[I(S)]$ as well. Consider the complexity of Algorithm 5, line 2 takes $O(\sigma)$ time. Line 3 and line 4 takes $O(\sigma)$ time, respectively. In line 5, the simulation of IC process in G'^T takes $O(\sigma)$ time. Therefore, the complexity of Algorithm 5 is $O(\sigma)$ in total.

C. IMM_2nd Algorithm

Based on the generation of RR sets, we design IMM_2nd algorithm to solve IM problem under second-order IC. IMM_2nd is an RIS based algorithm, and follows the two-step framework in Section III-D.

In Sampling phase, the size of $\mathcal{R}$ is determined as same as in IMM [22]. For brevity, we omit the description of the implementation and refer interested readers to [22] for details. While IMM_2nd generates each RR set under second-order IC by GenerateRRset_2ndIC (Algorithm 5). In Node Selection phase, IMM_2nd selects seed nodes following the standard greedy strategy. Specifically, it iteratively adds a selected node into seed set S_k, until $|S_k|=k$. In each iteration, it selects the node i with maximum $Cover_{\mathcal{R}}(\{i\})$, adds i into S_k, and deletes the RR sets satisfying $R_j \cap \{i\} \neq \emptyset$ in $\mathcal{R}$.

The size determination strategy of sampled RR sets and the node selection strategy of IMM_2nd are identical with IMM [22]. Although IMM_2nd samples RR sets in a different way, the $n \cdot Cover_{\mathcal{R}}(S)/|\mathcal{R}|$ is still an unbiased estimator of $\mathbb{E}[I(S)]$ under second-order IC. Therefore, IMM_2nd guarantees to obtain an $(1-\frac{1}{e}-\epsilon)$-approximate solution with at least $1-\frac{1}{n^l}$ probability according to Theorem 3, where l is a preset parameter positive correlated with the running time of the algorithm.

Theorem 3: [22] Given any $\epsilon > 0$, determining $|\mathcal{R}|$ by the sampling phase of IMM ensures that the node selection phase returns a $(1-\frac{1}{e}-\epsilon)$-approximate solution with at least $1-\frac{1}{n^l}$ probability.

As a RIS based algorithm, the complexity of IMM_2nd depends on the size of sampled RR sets, which is

$$O(\sum_{R_x \in \mathcal{R}} \sum_{i \in R_x} \sigma_i) = O(\overline{\sigma} \cdot |\mathcal{R}|). \tag{10}$$

Note that $\overline{\sigma}$ is the mean number of length two pathes of the nodes in sampled RR sets. This Compares with the complexity of IMM,

$$O(\sum_{R_x \in \mathcal{R}} \sum_{i \in R_x} d_i) = O(\overline{d} \cdot |\mathcal{R}|), \tag{11}$$

where $\overline{d}$ is the mean number of indegrees of the nodes in sampled RR sets.

D. Distributed IMM_2nd Algorithm

IMM_2nd algorithm may not be scalable to large networks, since the complexity increment of second-order IC model. We design a distributed extension of IMM_2nd algorithm, named DIMM_2nd, working on Hadoop for large networks. This algorithm follows the same two-step framework as IMM_2nd. When DIMM_2nd samples RR sets, it parallelly finishes the sampling works on distributed machines.

Specifically, each distributed machine copies the graph $G(V,E,P)$ at first. One central machine executes the main part of DIMM_2nd, and the other machines finish the Map-Reduce process, i.e., the RR sets sampling.

In Sampling phase, the central machine first determines the size of RR sets following the same strategy in IMM_2nd. It uniformly allocates the sampling tasks to each mapper. The mappers work by mapping over the sampled RR sets and emitting the key-value pairs to corresponding reducers. The key is the *id* of a node, and the value is the *id*s of the RR sets which contain the node. Each reducer merges the pairs with the same node *id*, and emit these pairs to the central machine. In Node Selection phase, the central machine finds the seed set by all recieved pairs using the same greedy strategy as IMM_2nd.

Combined with the example in Figure 2, a mapper samples the RR sets $R_1 = \{i,k,l\}$ and the other one samples $R_2 = \{i,k\}$. They create key-value pairs $\{<i,1>,<k,1>,<l,1>\}$ and $\{<i,2>,<k,2>\}$, respectively, and emit them. The reducer corresponding to node i receives $<i,1>,<i,2>$, merges them as $<i,[1,2]>$, and emits it to the central machine. The rest of the pairs are processed in the same way.

VI. Experimental Results

In this section, we perform comprehensive experimental evaluations on the developed second-order IC model, IMM_2nd and DIMM_2nd algorithms. The effectiveness of the developed second-order IC model is evaluated by the tweet cascade data from Twitter. The efficiencies of proposed IMM_2nd and DIMM_2nd algorithms are evaluated on five real social networks.

All programs are written in C++. The DIMM_2nd algorithm is performed on a Hadoop platform which contains 20 distributed machines with Intel(R) Core(TM) i3-2120 CPU 3.30 GHz and 12 GB memory. Other experiments are performed on a linux machine with 64G memory and Intel Xeon 3.0GHz CPU.

A. Effectiveness Evaluation of Second-Order IC

A node in the Twitter follower network represents a user and a directed edge (i,j) represents that user j follows user i. The Twitter follower network used in our experiments was crawled on April 2018. The network contains 801K nodes and 10.2M directed edges. We crawl the timeline of each user's homepage to obtain the tweet cascades. The cascade data from January 2018 to April 2018 are used in our experiments.

The tweet cascades reflect the influence diffusions between social network users. We should ensure the following principles when we estimate influence probabilities in Twitter: i) the influence probabilities are based on tweet cascades; ii) the influence probabilities are indepent with the time variation. Thus, we use the Bernoulli model estimate the influence probabilities [37].

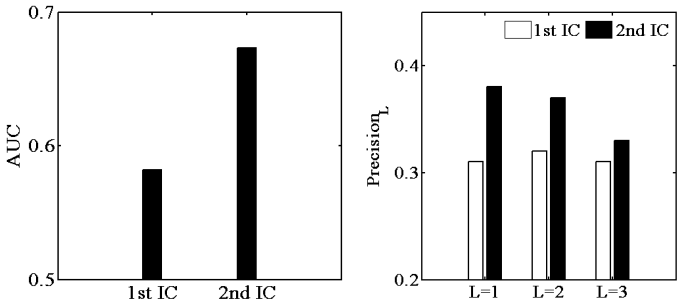
$$p_{i,j} = \frac{A_{i,j}}{A_i}, \tag{12}$$

where A_i is the number of tweets posted by node i, and $A_{i,j}$ is the number of tweets diffused from node i to j. The second-order influence probabilities are computed as following:

$$p_{i,j,k} = \frac{A_{i,j,k}}{A_i}, \tag{13}$$

where $A_{i,j,k}$ is the number of cascades through path (i,j,k). Tweet cascade data from February 2018 to March 2018 is used to compute the first-order and second-order influence probabilities.

We use the prediction accuracy of tweet cascades to evaluate the effectiveness of second-order IC model. In Twitter network, we randomly choose a source node as a seed. If there is at least one tweet cascade from source i to node k, it indicates that node k is influenced by source i. The cascade data in April 2018 are used as the ground truth to evaluate the predicted results.



(a) AUC (b) Precision

Fig. 5. Tweet cascade prediction on Twitter

Specifically, we use AUC (area under the ROC curve) and Precision to evaluate the accuracy [38]. AUC can be interpreted as the probability that a randomly chosen user influenced by the source is given a higher probability than a randomly chosen user not influenced by the source. Precision is defined as,

$$\text{Precision}_L = L'/L, \tag{14}$$

where L is the total number of predicted users, and L' is the number of users actually influenced by the source. We randomly pick 10^3 sources and report the average. Note that given a source i, if there is only one path (i,j) from i to j in the network, the probabilities that j is influenced by i under first-order IC and second-order IC are the same. So we delete this kind of out-neighbors of the source to exhibit the difference between two models.

As shown in Fig. 5(a), second-order IC model improves the AUC value by 15.6% compared to the IC model. Fig. 5(b) shows the Precision_L values, $L = 1,2,3$. Second-order IC model outperforms the IC model on the prediction of top-1, top-2 and top-3 influenced users. It improves the Precision_1 value by 22.58% compared to the IC model.

The real tweet cascades reflect influence diffusions among different users. Our proposed model takes the advantage of modeling the influence diffusion in a second-order way and its proper influence propagation strategies. Therefore, it has a higher accuracy of prediction.

B. Efficiency Evaluation of IMM_2nd Algorithm

In this part, we show the efficiency evaluations of proposed IMM_2nd and DIMM_2nd algorithms on five real social networks.

The chosen comparisons are two RIS based algorithms TIM [21], TIM+ [21], one greedy algorithm CELF++ [11] and one heuristic algorithm Max-Degree (MD) which chooses nodes with maximum out-neighbors as seed nodes. Because there is no existing algorithm solving IM problem under second-order IC model. We use Algorithm 3 to estimate the influence of some nodes set in CELF++, and use Algorithm 5 to generate RR sets in TIM and TIM+.

Five real social networks used in our experiments are shown in Table III. *NetHEPT*, *Epinions*, *DBLP* and *LiveJournal* are publicly available[1], while *Twitter* is the crawled dataset described in Section VI-A.

[1] https://snap.stanford.edu/data/

TABLE IV
EVALUATING THE EFFECT OF α

	Influence spread with $k=50$											
α	$\alpha=0$	$\alpha=0.1$	$\alpha=0.2$	$\alpha=0.3$	$\alpha=0.4$	$\alpha=0.5$	$\alpha=0.6$	$\alpha=0.7$	$\alpha=0.8$	$\alpha=0.9$	$\alpha=1$	ave.
NetHEPT	818.72	837.47	848.92	841.04	858.33	862.04	845.98	857.09	833.63	876.85	552.05	851.25
Epinions	3412.13	3490.25	3537.96	3505.11	3577.16	3592.65	3525.7	3572.02	3474.23	3654.37	2300.71	3547.72
DBLP	11698.81	11966.68	12130.25	12017.61	12264.66	12317.75	12088.19	12247.02	11911.72	12529.37	7888.21	12163.68
LiveJournal	76003.51	77743.74	78806.4	78074.61	79679.67	80024.52	78533.2	79565.02	77386.73	81399.37	51247.21	79023.66

TABLE III
STATISTICS OF REAL NETWORKS

Datasets	n	m	σ	Type
NetHEPT	15K	59K	600K	undirected
Epinions	76K	509K	39.9M	directed
DBLP	655K	2M	15.2M	undirected
Twitter	801K	10.2M	76.5M	directed
LiveJournal	4.8M	69M	1620.7M	directed

In *Twitter*, since we have the information of data flow, all influence probabilities are computed as the same in Section VI-A. In the other datasets, we only has the information of network structures. Thus, the first-order influence probabilities are computed following the conventional computation [12], [21], [22], [36],

$$p_{i,j}=\frac{1}{d_j}. \tag{15}$$

Because the second-order influence probability takes the previous activated nodes into consideration, we use the Autoregressive Model [34] to compute the second-order influence probabilities,

$$p_{i,j,k}=\frac{p'_{i,j,k}}{\sum_{l\in O_j} p'_{i,j,l}}, \tag{16}$$

where $p'_{i,j,k}=(1-\alpha)p_{j,k}+\alpha p_{i,k}$. The parameter $\alpha\in[0,1]$ is a constant to control the effect from the previous activated nodes. To determine the value of α, we vary the value of this parameter from 0 to 1, and evaluate its effect by testing the influence of the seed set obtained by IMM_2nd algorithm. Table IV shows the different influences of seed set whose size is 50 on different networks. The influence of seed set is quite uncommon when $\alpha=0$ and $\alpha=1$. Besides , $\alpha=0$ means there is no effect from the previous activated nodes, and $\alpha=1$ means any activation only depends on the previous activated nodes, which are both unreasonable. Therefore, we omit the results of $\alpha=0$ and $\alpha=1$, and compute the average influence with $\alpha=0.1$ to $\alpha=0.9$. When $\alpha=0.2$, the influence of seed set is mostly close to the average. Thus, we use $\alpha=0.2$ as a constant parameter in our experiments.

In the rest of experiments, we keep the preset parameters $\alpha=0.2$, $l=1$ and $\epsilon=0.1$ as a general setting. For the other parameters defined for particular algorithms, we take the recommended values in the corresponding papers if available. In each of our experiments, we run each method 5 times and report the average.

1) Influence of Seed Set: We compare the influences of seed sets returned by different algorithms under second-order IC model. The influence of any obtained seed set S_k is estimated by taking its average influence in 10000 simulations of the second-order IC process.

On small network, *NetHEPT* shown in Fig. 6(a), the seed set influences of all the algorithms are comparable with no significant difference, except the heuristic MD algorithm. The seed set influence of IMM_2nd is averagely 59.3% more than MD algorithm. Although MD has the least running time, it has the worst solution quality.

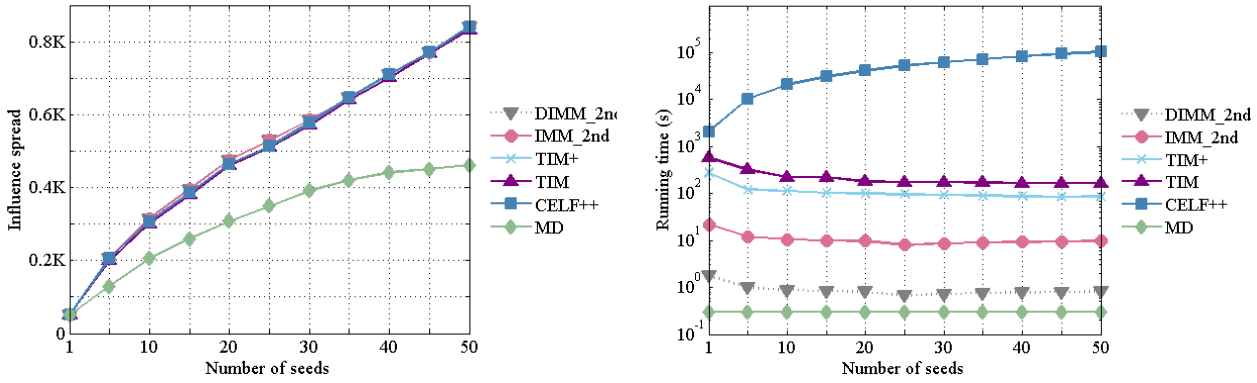
Fig. 7 illustrates the the seed set influences of different algorithms on large networks. Because of the time limit, CELF++ is not evaluated on *Twitter* and *LiveJournal*. The experimental results are similar with the result on small network. The seed set influences of all the algorithms are comparable with no significant difference, except the heuristic MD algorithm. The seed set influence of IMM_2nd is averagely 30.5% more than MD algorithm.

In large networks, experimental results shows that there is a significant influential user who strongly influences numerous users. This indicates that few key user can influence numerous users in real social networks.

2) Running Time: We examine the running time of each algorithm, by varying k from 1 to 50. Note that we only test the algorithms with at least 10^6 seconds running time, and omit the others since the time limit.

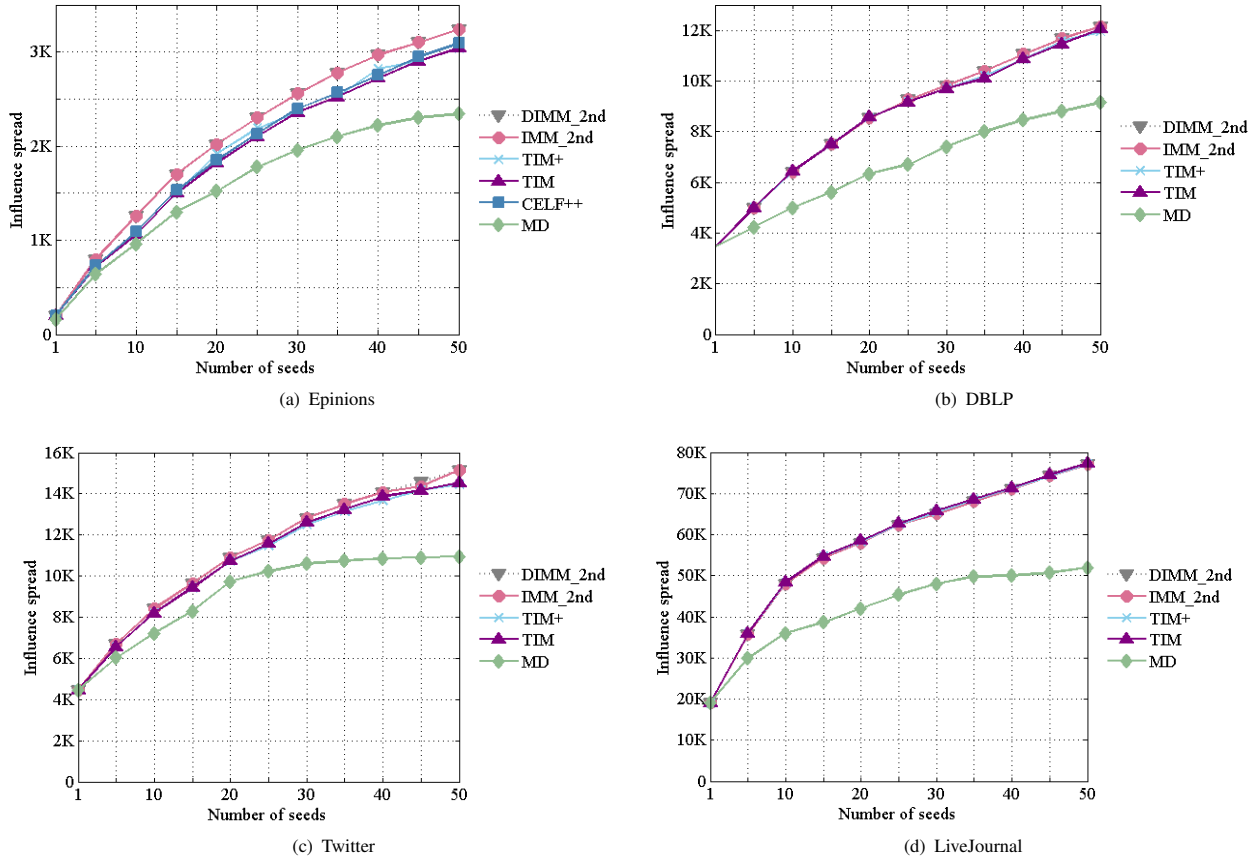
As shown in Fig. 6(b), our IMM_2nd and DIMM_2nd algorithms significantly outperform the other compared algorithms in *NetHEPT*, except the MD algorithm. The seed node influence of distributed DIMM_2nd algorithm is 59.3% more than MD algorithm by taking less than 1.7 times running time. In addition, the proposed distributed DIMM_2nd algorithm is 12.36 times faster than IMM_2nd algorithm. In terms of RIS based algorithms, TIM+ shows a better performance than TIM, because it adds an intermediate step that heuristically tighten $|\mathcal{R}|$, which leads to higher efficiency. IMM_2nd is 12.26 times faster than TIM+, since it has a smaller theoretical threshold of $|\mathcal{R}|$. The greedy algorithm CELF++ takes much longer running time than RIS based algorithms, which is consistent with the analysis in Section III-C.

On large networks, as shown in Fig. 8, IMM_2nd and DIMM_2nd algorithms significantly outperform the other compared algorithms on each dataset as well. It shows that CELF++ is not scalable to the large networks and TIM may not be suitable for the networks larger than *LiveJournal*. On each networks, DIMM_2nd algorithm always achieves a high efficiency, which is suitable for large networks. It overcomes the complexity increment of second-order IC model, and is averagely 12.86 times faster than IMM_2nd.



(a) Influence spread (b) Running time

Fig. 6. Evaluations of influence spread and running time on small network with $k=1$ to 50, $\alpha=0.2$, $l=1$ and $\epsilon=0.1$



(a) Epinions (b) DBLP

(c) Twitter (d) LiveJournal

Fig. 7. Influence spread on large networks with $k=1$ to 50, $\alpha=0.2$, $l=1$ and $\epsilon=0.1$

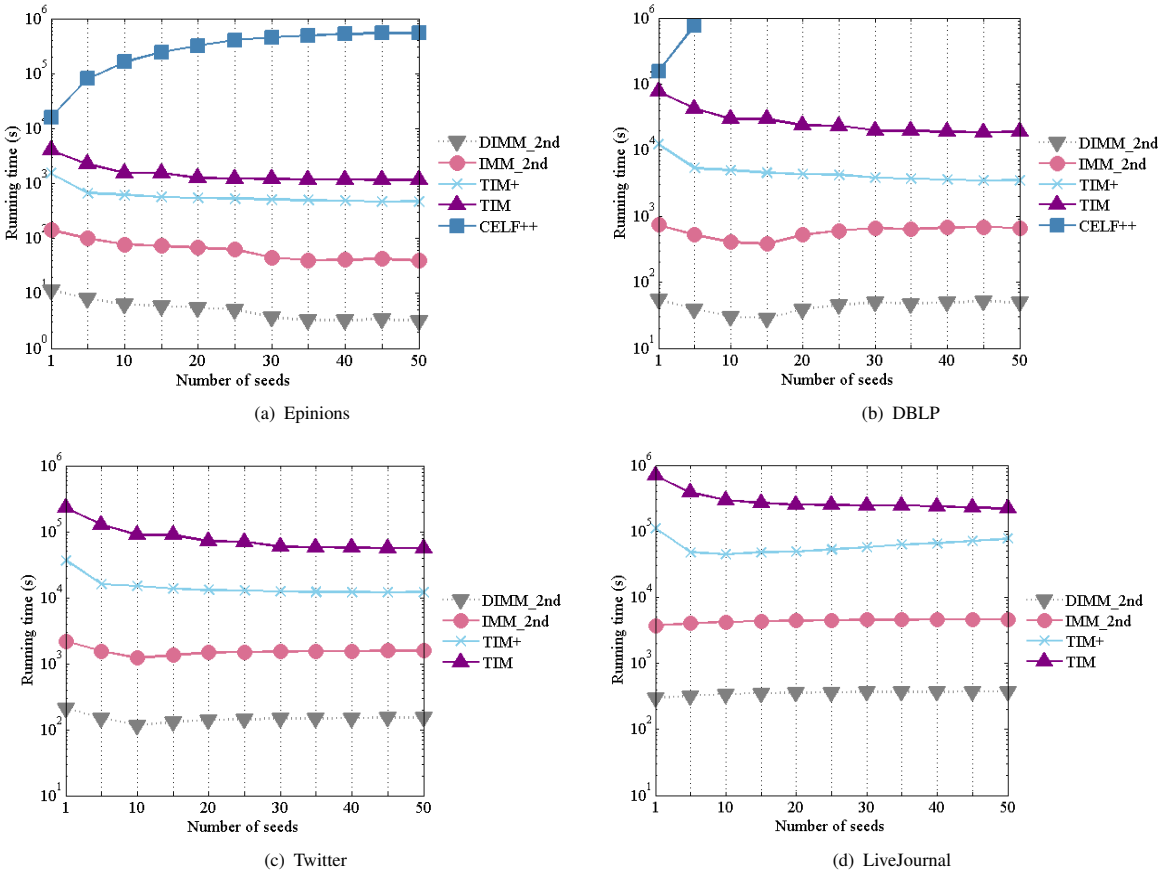
VII. Conclusions

Designing effective diffusion models and efficient algorithms for the IM are pivotal and challenging. The existing research of IM only considers influences from nodes' direct in-neighbors, losing the information of previous activations. Therefore, this paper investigates the second-order IM. We propose the second-order IC model which takes previous activations into consideration. Furthermore, we present the relation between IC model and second-order IC model, and prove that IM under the second-order IC is a NP-Hard problem. Consequently, we design an efficient RIS based algorithm and its distributed extension for the IM. Extensive experimental results demonstrate that the second-order IC model significantly improves the simulation accuracy of influence diffusions, and the developed algorithms are efficient.

References

[1] P. Domingos and M. Richardson, "Mining the network value of customers," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2001, pp. 57–66.

[2] A. Kuhnle, M. A. Alim, X. Li, H. Zhang, and M. T. Thai, "Multiplex influence maximization in online social networks with heterogeneous diffusion models," *IEEE Transactions on Computational Social Systems*, vol. 5, no. 2, pp. 418–429, 2018.



(a) Epinions (b) DBLP

(c) Twitter (d) LiveJournal

Fig. 8. Running time on large networks with $k=1$ to 50, $\alpha=0.2$, $l=1$ and $\epsilon=0.1$

[3] J. Li, Z. Cai, M. Yan, and Y. Li, "Using crowdsourced data in location-based social networks to explore influence maximization," in *35th Annual IEEE International Conference on Computer Communications, INFOCOM 2016, San Francisco, CA, USA, April 10-14, 2016*, 2016, pp. 1–9.

[4] T. Shi, S. Cheng, Z. Cai, Y. Li, and J. Li, "Retrieving the maximal time-bounded positive influence set from social networks," *Personal and Ubiquitous Computing*, vol. 20, no. 5, pp. 717–730, 2016.

[5] M. Han, M. Yan, Z. Cai, Y. Li, X. Cai, and J. Yu, "Influence maximization by probing partial communities in dynamic online social networks," *Trans. Emerging Telecommunications Technologies*, vol. 28, no. 4, 2017.

[6] Z. He, Z. Cai, J. Yu, X. Wang, Y. Sun, and Y. Li, "Cost-efficient strategies for restraining rumor spreading in mobile social networks," *IEEE Trans. Vehicular Technology*, vol. 66, no. 3, pp. 2789–2800, 2017.

[7] Z. He, Z. Cai, and X. Wang, "Modeling propagation dynamics and developing optimized countermeasures for rumor spreading in online social networks," in *35th IEEE International Conference on Distributed Computing Systems, ICDCS 2015, Columbus, OH, USA, June 29 - July 2, 2015*, 2015, pp. 205–214.

[8] K. Zia, D. K. Saini, A. Muhammad, and A. Ferscha, "Nature-inspired computational model of population desegregation under group leaders influence," *IEEE Transactions on Computational Social Systems*, vol. 5, no. 2, pp. 532–543, 2018.

[9] N. Alduaiji, A. Datta, and J. Li, "Influence propagation model for clique-based community detection in social networks," *IEEE Transactions on Computational Social Systems*, vol. 5, no. 2, pp. 563–575, 2018.

[10] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2003, pp. 137–146.

[11] A. Goyal, W. Lu, and L. V. Lakshmanan, "Celf++: optimizing the greedy algorithm for influence maximization in social networks," in *Proceedings of the 20th international conference companion on World wide web*. ACM, 2011, pp. 47–48.

[12] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance, "Cost-effective outbreak detection in networks," in *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2007, pp. 420–429.

[13] W. Chen, Y. Wang, and S. Yang, "Efficient influence maximization in social networks," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2009, pp. 199–208.

[14] W. Chen, C. Wang, and Y. Wang, "Scalable influence maximization for prevalent viral marketing in large-scale social networks," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2010, pp. 1029–1038.

[15] W. Chen, Y. Yuan, and L. Zhang, "Scalable influence maximization in social networks under the linear threshold model," in *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 2010, pp. 88–97.

[16] Y. Wang, G. Cong, G. Song, and K. Xie, "Community-based greedy algorithm for mining top-k influential nodes in mobile social networks," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2010, pp. 1039–1048.

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

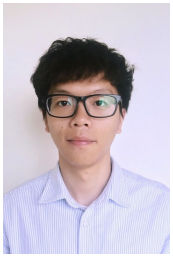
[18] K. Jung, W. Heo, and W. Chen, "Irie: Scalable and robust influence maximization in social networks," in *Data Mining (ICDM), 2012 IEEE 12th International Conference on*. IEEE, 2012, pp. 918–923.

[19] J. Kim, S.-K. Kim, and H. Yu, "Scalable and parallelizable processing of influence maximization for large-scale social networks?" in *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*. IEEE, 2013, pp. 266–277.

[20] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier, "Maximizing social influence in nearly optimal time," in *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*. SIAM, 2014, pp. 946–957.

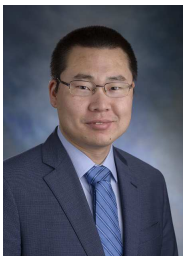
[21] Y. Tang, X. Xiao, and Y. Shi, "Influence maximization: Near-optimal time complexity meets practical efficiency," in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. ACM, 2014, pp. 75–86.

[22] Y. Tang, Y. Shi, and 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.
[23] N. Du, L. Song, M. G. Rodriguez, and H. Zha, "Scalable influence estimation in continuous-time diffusion networks," in *Advances in neural information processing systems*, 2013, pp. 3147–3155.
[24] M. G. Rodriguez, D. Balduzzi, and B. Schölkopf, "Uncovering the temporal dynamics of diffusion networks," *arXiv preprint arXiv:1105.0697*, 2011.
[25] M. Gomez-Rodriguez, J. Leskovec, and B. Scholkopf, "Modeling information propagation with survival theory," in *International Conference on Machine Learning*, 2013, pp. 666–674.
[26] M. G. Rodriguez and B. Schölkopf, "Influence maximization in continuous time diffusion networks," *arXiv preprint arXiv:1205.1682*, 2012.
[27] D. Kempe, J. Kleinberg, and É. Tardos, "Influential nodes in a diffusion model for social networks," in *International Colloquium on Automata, Languages, and Programming*. Springer, 2005, pp. 1127–1138.
[28] L. Seeman and Y. Singer, "Adaptive seeding in social networks," in *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*. IEEE, 2013, pp. 459–468.
[29] N. Chen, "On the approximability of influence in social networks," *Siam Journal on Discrete Mathematics*, vol. 23, no. 3, pp. 1400–1415, 2010.
[30] Y. Zhu, D. Li, R. Yan, W. Wu, and Y. Bi, "Maximizing the influence and profit in social networks," *IEEE Transactions on Computational Social Systems*, vol. 4, no. 3, pp. 54–64, 2017.
[31] N. Barbieri, F. Bonchi, and G. Manco, "Topic-aware social influence propagation models," in *Data Mining (ICDM), 2012 IEEE 12th International Conference on*. IEEE, 2012, pp. 81–90.
[32] H. Zhang, T. N. Dinh, and M. T. Thai, "Maximizing the spread of positive influence in online social networks," in *IEEE International Conference on Distributed Computing Systems*, 2013, pp. 317–326.
[33] T. Zhou, J. Cao, B. Liu, S. Xu, Z. Zhu, and J. Luo, "Location-based influence maximization in social networks," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 2015, pp. 1211–1220.
[34] Y. Wu, Y. Bian, and X. Zhang, "Remember where you came from: on the second-order random walk based proximity measures," *Proceedings of the Vldb Endowment*, vol. 10, no. 1, pp. 13–24, 2016.
[35] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functionsłi," *Mathematical Programming*, vol. 14, no. 1, pp. 265–294, 1978.
[36] H. T. Nguyen, M. T. Thai, and T. N. Dinh, "Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks," in *Proceedings of the 2016 International Conference on Management of Data*. ACM, 2016, pp. 695–710.
[37] A. Goyal, F. Bonchi, and L. V. Lakshmanan, "Learning influence probabilities in social networks," in *Proceedings of the third ACM international conference on Web search and data mining*. ACM, 2010, pp. 241–250.
[38] L. L and T. Zhou, "Link prediction in complex networks: A survey," *Physica A Statistical Mechanics and Its Applications*, vol. 390, no. 6, pp. 1150–1170, 2010.



Wenyi Tang received the B.S. degree from the School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu, China, in 2015. He is currently a Ph.D. student in the School of Computer Science and Engineering, University of Electronic Science and Technology of China. His research interests include social networks and data mining.

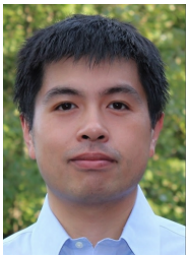
Guangchun Luo received the Ph.D. degree in computer science from University of Electronic Science and Technology of China, Chengdu, China, in 2004. He is currently a professor and the Associate Dean of computer science at the UESTC. He has published over sixty journal and conference papers in his fields. His research interests include computer networks and big data.



Yubao Wu received the Ph.D. in Computer Science from Case Western Reserve University. Dr. Wu is currently an assistant professor in the Department of Computer Science at Georgia State University. His research interests include big data analytics, data mining, and bioinformatics.

Ling Tian received the bachelor, master and Ph.D. degrees from the school of computer science, University of Electronic Science and Technology of China in 2003, 2006 and 2010, respectively. She is currently an Associate Professor in UESTC. She had been a visiting scholar in Georgia State University during 2013 in United States. Her research interests include image and video coding, signal processing.

Xu Zheng received his B.S. and M.S. degree from School of Computer Science and Technology at Harbin Institute of Technology. Mr. Zheng is currently an assistant professor in School of Computer Science and Engineering, University of Electronic Science and Technology of China, and a PhD student in the Department of Computer Science at Georgia State University. Mr. Zheng's research areas focus on wireless network and Big Data.



Zhipeng Cai received his Ph.D. and M.S. degrees in the Department of Computing Science at University of Alberta, and B.S. degree from Beijing Institute of Technology. Dr. Cai is currently an Associate Professor in the Department of Computer Science at Georgia State University. Dr. Cai's research areas focus on Networking, Privacy and Big data. He has published more than 50 journals papers, including more than 20 IEEE/ACM Transactions papers, such as IEEE Transactions on Knowledge and Data Engineering, IEEE Transactions on Dependable and Secure Computing, IEEE/ACM Transactions on Networking, IEEE Transactions on Mobile Computing, etc. Dr. Cai is the recipient of an NSF CAREER Award. He is an editor/guest editor for Algorithmica, Theoretical Computer Science, Journal of Combinatorial Optimization, and IEEE/ACM Transactions on Computational Biology and Bioinformatics. He is a senior member of the IEEE.