Social Influence Maximization in Hypergraph in Social Networks

Jianming Zhu, Junlei Zhu, Smita Ghosh, Weili Wu, *Member, IEEE*, and Jing Yuan, *Student Member, IEEE*

Abstract—Crowd psychology plays an important role in determining the kind of activities that a person performs. In reality, in a social network, crowd influence has been observed and it cannot be ignored when considering information diffusion problems. In this paper, we model crowd influence as a hyperedge $e = (H_e, v)$ with weight $0 \leq P_e \leq 1$, where H_e is the head node set and v is the tail node, means v will be activated by H_e with probability P_e only after each node in H_e is activated. Then, the Social Influence Maximization Problem in Hypergraph (SIMPH) aims to select k initially-influenced seed users in a directed hypergraph $G = (V, E, P)$. The objective is to maximize the expected number of eventually-influenced users. We show that SIMPH is NP-hard and the objective function is neither submodular nor supermodular. We develop a lower bound and an upper bound that are submodular. We prove that maximizing these two bounds are still NP-hard under IC model. Then, we present a D-SSA algorithm for general weighted social influence maximization problem preserving $(1 - 1/e - \epsilon)$-approximation. We formulate a sandwich approximation framework, which preserves a theoretical analysis result. Finally, we evaluate our algorithm on real world data sets. The results show the effectiveness and the efficiency of the proposed algorithm.

Index Terms—Social influence maximization, independent cascade, crowd influence, hyperpraph, sandwich approximation framework

1 Introduction

Influenc maximization problem aims to select k initially-influenced seed users to maximize the expected number of eventually-influenced users as it has received tremendous attention in the last few decades. Influence maximization finds its application in many domains, such as viral marketing [1], epidemic control and assessing cascading failures within complex systems. In this paper, we extend the Social Influence Maximization Problem by considering the crowd influence in hypergraphs.

It has been shown that individual behavior is heavily influenced by the crowd [2]. The crowd influence is different from the combined independent influences of people in the crowd. It surpasses the combination of the independent influence from each person in the crowd. Below is a specific example. Directed edges represent the influence from A or B to C. The influences C receives from A and B are independent of each other. According to the crowd psychology, if both A and B are active, there should exist a crowd influence in addition to A's and B's influences. A hyperedge is used to depict such a crowd influence. In this paper, directed hyperedge is represented as $e = (H_e, v)$ where H_e is the head set and v is the tail. As shown in Fig. 1, e_3 is a hyperedge where $H_e = \{A, B\}$ and C is tail. The weight of e_3 means the crowd influence from $\{A, B\}$ to C is 0.7.

This phenomenon leads to non-submodularity when considering the social influence maximization problem. The non-submodularity comes from the crowd psychology. The crowd psychology reveals that the crowd influence is different from the combined independent influences of people in the crowd. This phenomenon yields non-submodularity in social influence propagations, which are modeled through hypergraphs. In this paper, information diffusion is based on independent cascade (IC) model. As shown in Fig. 1, there are three independent events: A activate C with probability 0.5, B activate C with probability 0.4 and the crowd influence from A and B activate C with probability 0.7. Then the activated probability of C is $1 - (1 - 0.5)(1 - 0.4)(1 - 0.7) = 0.91$.

Note that influences through hypergraphs are not submodular, we cannot adapt existing social influence maximization methods to solve the SIMPH. Therefore, challenges are posed to solve the SIMPH. The first challenge is to deal with the non-submodularity. The problem hardness and approximability need to be explored. New algorithms are needed, since a simple greedy algorithm can no longer guarantee an approximation ratio. Another challenge is the scalability. Since hyperedges change the scalability, it is difficult to reduce their complexities.

- *J. Zhu is with the School of Engineering Science, University of Chinese Academy of Sciences, 19A Yuquan Rd., Beijing 101408, China. E-mail: jmzhu@ucas.ac.cn.*
- *J. Zhu is with Jiaxing University, Zhejiang 314001, China. E-mail: zhujl-001@163.com.*
- *S. Ghosh and J. Yuan are with the University of Texas at Dallas, Richardson, TX 75080. E-mail: {smita.ghosh1, jing.yuan}@utdallas.edu.*
- *W. Wu is with the Taiyuan Institute of Technology, Taiyuan, Shanxi 030003, China, and the University of Texas at Dallas, Richardson, TX 75080. E-mail: weiliwu@utdallas.edu.*

Manuscript received 14 June 2018; revised 10 Sept. 2018; accepted 27 Sept. 2018. Date of publication 8 Oct. 2018; date of current version 3 Dec. 2019. (Corresponding author: Jianming Zhu.)
Recommended for acceptance by V. S. Borkar.
Digital Object Identifier no. 10.1109/TNSE.2018.2873759

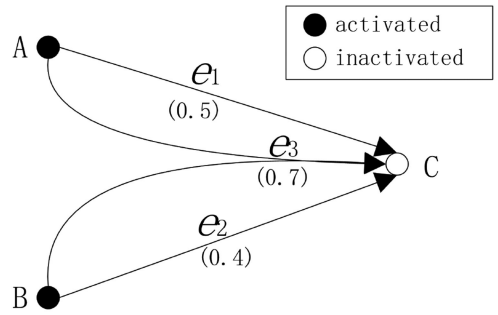


Fig. 1. An example of hyperedge.

1.1 Related Works

Kempe et al. [3] were the first to formulate SIMP as an optimization problem under the IC model. They prove SIMP to be NP-hard under IC model and design a natural greedy algorithm that yields $(1 - 1/e - \epsilon)$-approximate solutions for any $\epsilon > 0$. Motivated by this celebrate work, a fruitful literature for SIMP ([4], [5]) have been developed. However, most of the existing methods are either too slow for billion-scale networks such as Facebook, Twitter and World Wide Web or fail to retain the $(1 - 1/e - \epsilon)$-approximation guarantees.

TIM/TIM+ [6] and IMM [7] are two scalable methods with $(1 - 1/e - \epsilon)$-approximation guarantee for SIMP. Tang et al. [6], [7] utilize a novel RIS sampling technique introduced by Borgs et al. [8]. TIM+ and IMM attempt to generate a $(1 - 1/e - \epsilon)$-approximate solution with minimal numbers of RIS samples. However, they may take days on billion-scale networks.

Later, Nguyen et al. [9] make a breakthrough and proposed two novel sampling algorithms SSA and D-SSA. Unlike the previous heuristic algorithms, SSA and D-SSA are faster than TIM+ and IMM while providing the same $(1 - 1/e - \epsilon)$-approximate guarantee. SSA and D-SSA are the first approximation algorithms that use minimum numbers of samples, meeting strict theoretical thresholds characterized for SIMP.

Although there are a large amount of literatures for SIMP, almost all of SIMP are submodular. Few results [10] are provided when the influence propagation model even slightly violates the submodularity. Note that influences through hypergraphs are not submodular, we cannot adapt existing social influence maximization methods to solve the SIMPH. The latest approach is based on the sandwich [11] approximation strategy, which approximates the objective function by looking for its lower bound and upper bound.

1.2 Contributions

Our contributions are summarized as follows:

1) Motivated by the crowd influence in the social networks, we propose the Social Influence Maximization Problem in Hypergraph (SIMPH) that aims to maximize the expected number of eventually-influenced users under independent cascade model.
2) We assess the challenges of the proposed maximization problem by analyzing computational complexity and properties of objective function. First, we show the SIMPH is NP-hard under IC model. Moreover, the objective function of this problem is proved neither submodular nor supermodular.
3) To achieve practical approximate solution, we develop a lower bound and upper bound of objective function. We prove that maximizing these two bounds are still NP-hard under IC model. However, we also prove that both lower bound and upper bound are submodular. Motivated by RIS sampling method, we present a D-SSA algorithm for general weighted social influence maximization problem. Additionally, D-SSA preserves $(1 - 1/e - \epsilon)$-approximation.
4) For solving SIMPH, first we develop a randomized algorithm for estimation of the objective function in SIMPH. Second, based on influence increment maximization, a greedy strategy is presented. Finally, We formulate a sandwich approximation framework, which preserves a theoretical analysis result.
5) Last, we verify our algorithm on real world data sets. The results show the effectiveness and the efficiency of the proposed algorithm.

TABLE 1
Frequently Used Notation

Notation	Description
$G = (V, E, P)$	A social network, where V is the node set, E is edge set. Each edge is associated with an influence probability P.
$G = (V, C, E, P, f)$	A social network, where V is the node set, E is edge set. Each edge is associated with an influence probability P. $C \subseteq V$ is a candidate seed set. f is weight function of node.
$e = (H_e, v)$	e is a directed hyperedge when $\|H_e\| \geq 2$, while e is a normal directed edge when $\|H_e\| = 1$. H_e is called the head and v is called the tail.
P_e	influence probability on edge e
$m = \|E\|$	the number of edges in G
$n = \|V\|$	the number of nodes in G
k	the number of seeds
$\sigma(S)$	The expected number of eventually-influenced nodes with initial seed set S in social network G under diffusion model.

The rest of this paper is organized as follows. In Section 2, we formulate the Social Influence Maximization Problem in Hypergraph. The statement of NP-hardness and properties of objective function will be given in Section 3. In Section 4, we develop a lower bound and upper bound. Algorithms for solving SIMPH are designed in Section 5. Experiment results are shown in Section 6 and we draw a conclusion in Section 7. Table 1 summarizes the frequently used symbols and their meaning.

2 Problem Formulation

In this section, we first review independent cascade model, and then present the statement of the Social Influence Maximization Problem in Hypergraph.

2.1 Independent Cascade Model

The Independent Cascade model [3] is the most widely used information diffusion model. The Social Influence Maximization Problem in Hypergraph is based on IC model.

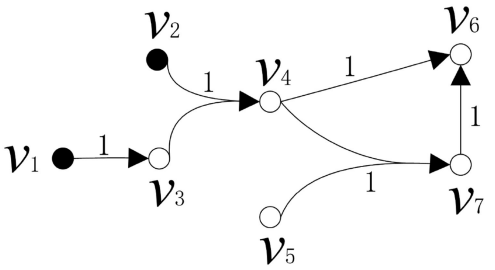


Fig. 2. An example of information diffusion process with initial seeds $\{V_1, V_2\}$.

Given a directed hypergraph $G = (V, E, P)$, where V is a set of nodes(i.e., users in an OSN), E is a set of directed hyperedges and P is the weight function on hyperedge set E. Hyperedges represent influence propagation directions, including personal and crowd influences. For a hyperedge $e = (H_e, v)$, let H_e denote its head set of nodes and v be the tail node. If H_e contains only one node u, it means e is a normal directed edge and the influence is personal. While H_e contains more than one node, the hyperedge e means there is crowd influence from H_e to v. Let P_e denote the weight of e, representing the influence propagation probability ($0 \leq P_e \leq 1$). Specifically, P_e is the probability that v is activated by H_e after each node in H_e is activated.

IC model assumes a seed set $S \subseteq V$. Let S_t be the nodes that are activated in step $t (t = 0, 1, \ldots)$ and $S_0 = S$. For hyperedge $e = (H_e, v)$, e is activated for the first time at step t only if $H_e \subseteq S_t$ and $H_e \setminus S_{t-1} \neq \emptyset$. The diffusion process is as follows. At step t, each activated hyperedge $e = (H_e, v)$ for the first time has only one chance to activate the inactivated node v with the probability of P_e. Note that a hyperedge e could only propagate the influence when all nodes in H_e first become all active. An example is shown in Fig. 2 to explain the diffusion process for SIMPH, where there are 7 nodes and the influence probability of each edge is 1. At the beginning, v_1 and v_2 are selected as initial seeds. At the first time step, v_3 will be activated by v_1. At the second time step, hyperedge $(\{v_2, v_3\}, v_4)$ is activated since v_2 and v_3 are both activated, then v_4 will will be activated by this hyperedge. At the third time step, v_6 will be activated by v_4. v_7 can not be activated since v_5 is inactive. Finally, $\{v_1, v_2, v_3, v_4, v_6\}$ are activated.

2.2 Influence Maximization in Hypergraph

The Social Influence Maximization Problem in Hypergraph also considers information diffusion in social network with crowd influence under the IC model. Given a directed hypergraph $G = (V, E, P)$, the objective is to select k initially-influenced seed users to maximize the expected number of eventually-influenced users

$$\max \sigma(S) \tag{1}$$

$$s.t. |S| \leq k. \tag{2}$$

Where S is the initial seed set and $\sigma(S)$ the expected number of eventually-influenced nodes.

3 Properties of Influence Maximization in Hypergraph

In this section, we first present statement of the hardness of the social influence maximization problem in hypergraph. Then discuss the properties of the objective function $\sigma(\cdot)$.

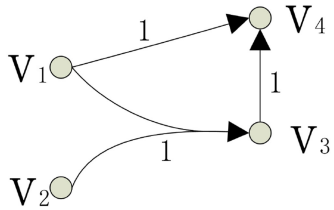


Fig. 3. Counter example.

3.1 Hardness Results

It is known that any generalization of a NP-hard problem is also NP-hard. The social influence maximization problem in a normal graph [3] has been proved NP-hard, which is a special case of our problem when the head set H_e just contains one node. Therefore, the SIMPH is obvious NP-hard.

Theorem 3.1. *The Social Influence Maximization Problem in Hypergraph is NP-hard.*

Also, we can get the following result of computing $\sigma(S)$ since it was proved #P-hard under the IC model in normal social network without considering crowd influences [3].

Theorem 3.2. *Given a seed node set S, computing $\sigma(S)$ is #P-hard under the IC model.*

3.2 Modularity of Objective Function

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

Theorem 3.3. *$\sigma(\cdot)$ is not submodular under IC model.*

Proof. We prove by a counter example. Consider Fig. 3. A social network $G = (V, E, P)$ has $V = \{v_1, v_2, v_3, v_4\}$ $E = \{(v_1, v_4), (v_3, v_4), (\{v_1, v_2\}, v_3)\}$ and $\{P_{(v_1,v_4)} = 1, P_{(v_3,v_4)} = 1, P_{(\{v_1,v_2\},v_3)} = 1\}$. Let $A = \emptyset$ and $B = \{v_2\}$, we have $\sigma(A) = 0, \sigma(B) = 1$. Putting v_1 into A and B, we have $\sigma(\{v_1\}) = 2$ and $\sigma(\{v_2, v_1\}) = 4$. Thus,

$$\sigma(A \cup \{v_1\}) - \sigma(A) \; < \; \sigma(B \cup \{v_1\}) - \sigma(B).$$

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

From the proof, we can see the reason why $\sigma(\cdot)$ is not submodular is the crowd influence from the newly added node and the existing seed nodes.

Theorem 3.4. *$\sigma(\cdot)$ is not supermodular under IC model.*

Proof. We prove by a counter example. Consider Fig. 3. Let $A = \emptyset$ and $B = \{v_1\}$, we have $\sigma(A) = 0, \sigma(B) = 2$. Putting v_3 into A and B, we have $\sigma(\{v_3\}) = 2$ and $\sigma(\{v_1, v_3\}) = 3$. Thus,

$$\sigma(A \cup \{v_3\}) - \sigma(A) \; > \; \sigma(B \cup \{v_3\}) - \sigma(B).$$

Therefore, $\sigma(\cdot)$ is not supermodular. □

4 Lower Bound and Upper Bound

There is no general method to optimize a non-submodular function. Lu et al. [11] proposed a sandwich approximation strategy, which approximates the objective function by looking for its lower bound and upper bound. In this

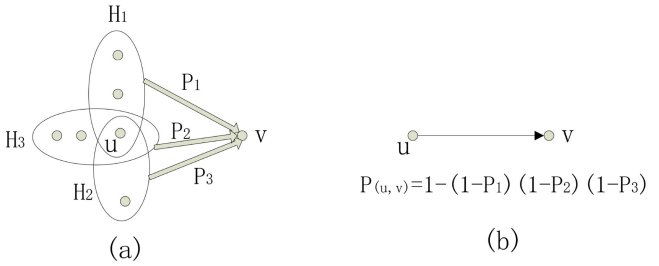


Fig. 4. An example for generation of each pair of nodes for the upper bound.

section, we will give a lower bound and an upper bound on $\sigma(\cdot)$. Then, we will analysis the properties of the lower bound and upper bound.

4.1 The Upper Bounds

A straight way to get an upper bound is to duplicate the influence of hyperedge in order to delete the hyperedges. An auxiliary problem $G_U = (V, E_U, P^U)$ is generated as follows. V is the same as nodes in the directed hypergraph G, E_U is a set of directed edges. For any two nodes u and v, if there exists a directed hyperedge $e = (H_e, v)$ that $u \in H_e$, then u connects to v in G_U. For each edge $(u, v) \in E_U$, suppose u appears in k head sets of hyperedges $(H_1, v), (H_2, v) \ldots, (H_k, v)$ in G. Then consider that k events "H_i influence v" are independent. Define a new influence probability of (u, v) to be $P^U_{(u,v)} = 1 - \prod_{i=1}^{i=k}(1 - P_{(H_i,v)})$. Then $G_U = (V, E_U)$ is a normal directed graph with influence probability $P^U_{(u,v)}$ on each directed edge. Find k initially-influenced seed users to maximize the expected number of eventually-influenced users in this auxiliary problem under IC model is defined as follows.

$$\max \sigma_U(S) \tag{3}$$

$$s.t. |S| \leq k. \tag{4}$$

Where S is the initial seed set and $\sigma_U(S)$ is the expected number of eventually-influenced nodes. The following theorem is true.

Theorem 4.1. *Given $G = (V, E, P)$, $\sigma_U(\cdot)$ is an upper bound of $\sigma(\cdot)$.*

Proof. We need to prove $\sigma_U(S) \geq \sigma(S)$ for any $S \in V$. According to the IC model, assume S_t and S'_t are the activated nodes in G and G_U respectively at step t. We only need to prove $S_t \subseteq S'_t$ at each time step.

When $t = 0$, $S_0 = S'_0 = S$. First, we will prove $S_1 \subseteq S'_1$ after the first time step. For each inactivated node v, we will prove that the total influence probability $P^U(v)$ in G_U is bigger than $P(v)$ in G. Assume v is the tail of l edges $(H_1, v), (H_2, v), \ldots, (H_l, v)$ in G. H_i will try to activate v with probability $P_{(H_i,v)}$ only if $H_i \subseteq S$. Then v can be activated with the probability $P(v) = 1 - \prod_{H_i \subseteq S}(1 - P_{(H_i,v)})$. On the other hand, for each hyperedge (H_i, v), the influence probability $P_{(H_i,v)}$ is duplicated to every node w in H_i to v according to the formulation process of upper bound graph. Then, $P^U(v) = 1 - \prod_{w \in S \cap (H_1 \cup H_2 \cup \cdots \cup H_l)} (1 - P_{(w,v)})$. For each $H_i \subseteq S$, $1 - P_{(H_i,v)} \geq (1 - P_{(H_i,v)})^{|H_i|}$. We have $P^U(v) = 1 - \prod_{w \in S \cap (H_1 \cup H_2 \cup \cdots \cup H_l)}(1 - P_{(w,v)}) = 1 - \prod_{w \in (S \cap H_1) \cup (S \cap H_2) \cup \cdots \cup (S \cap H_l)}(1 - P_{(w,v)}) \geq 1 - \prod_{w \in \cup_{H_i \subseteq S} H_i}(1 - P_{(w,v)}) = 1 - \prod_{H_i \subseteq S}(1 - P_{(H_i,v)})^{|H_i|} \geq 1 - \prod_{H_i \subseteq S}(1 - P_{(H_i,v)}) = P(v)$. Then, for each inactivated node v, $P^U(v) \geq P(v)$ means $S_1 \subseteq S'_1$ after the first time step.

Second, suppose $S_t \subseteq S'_t$ at step t, we will prove $S_{t+1} \subseteq S'_{t+1}$ after one time step. For each inactivated node v, we will prove that the total influence probability $P^U(v)$ in G_U is bigger than $P(v)$ in G. Assume v is the tail of l edges $(H_1, v), (H_2, v), \ldots, (H_l, v)$ in G. H_i will try to activate v with probability $P_{(H_i,v)}$ only if $H_i \subseteq S_t$. Then v can be activated with the probability $P(v) = 1 - \prod_{H_i \subseteq S_t}(1 - P_{(H_i,v)})$. On the other hand, for each hyperedge (H_i, v), the influence probability $P_{(H_i,v)}$ is duplicated to every node w in H_i to v according to the formulation process of upper bound graph. Then, $P^U(v) = 1 - \prod_{w \in S'_t \cap (H_1 \cup H_2 \cup \cdots \cup H_l)}(1 - P_{(w,v)})$. For each $H_i \subseteq S_t$, $1 - P_{(H_i,v)} \geq (1 - P_{(H_i,v)})^{|H_i|}$. We have $P^U(v) = 1 - \prod_{w \in S'_t \cap (H_1 \cup H_2 \cup \cdots \cup H_l)}(1 - P_{(w,v)}) = 1 - \prod_{w \in (S'_t \cap H_1) \cup (S'_t \cap H_2) \cup \cdots \cup (S'_t \cap H_l)}(1 - P_{(w,v)})$. Since $S_t \subseteq S'_t$, then $P^U(v) \geq 1 - \prod_{w \in (S_t \cap H_1) \cup (S_t \cap H_2) \cup \cdots \cup (S_t \cap H_l)}(1 - P_{(w,v)}) \geq 1 - \prod_{w \in \cup_{H_i \subseteq S_t} H_i}(1 - P_{(w,v)}) = 1 - \prod_{H_i \subseteq S_t}(1 - P_{(H_i,v)})^{|H_i|} \geq 1 - \prod_{H_i \subseteq S_t}(1 - P_{(H_i,v)}) = P(v)$. Then, for each inactivated node v, $P^U(v) \geq P(v)$ means $S_{t+1} \subseteq S'_{t+1}$ after the one time step. □

Fig. 4 shows an example for node pair u and v. Assume there are three head node sets H_1, H_2, H_3 contain u as shown in hypergraph (a), then (b) shows the generation process for directed edge (u, v) with probability $P_{(u,v)} = 1 - (1 - P_1)(1 - P_2)(1 - P_3)$.

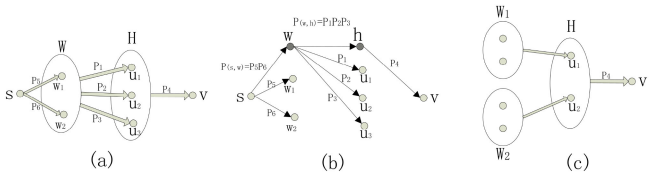


Fig. 5. An example for generation of directed graph for lower bound problem.

4.2 The Lower Bounds

Next, we will formulate a lower bound for SIMPH. The main idea is to delete some hyperedges from G, and only keep such hyperedge whose nodes in head set can be activate at the same time. That means all node in this hyperedge's head set have a same head set. As shown in Fig. 5a, for hyperedge $(H, v) = (\{u_1, u_2, u_3\}, v)$, there exist three hyperedges $(W, u_1), (W, u_2), (W, u_3)$ which have the same head set W that means u_1, u_2, u_3 will be activated at the same time. Such hyperedge (H, v) will keep in G, otherwise will be deleted such as shown in Fig. 5c.

Given an original SIMPH $G = (V, E, P)$, for each hyperedge $e = (H_e, v)$, suppose $H_e = \{u_1, u_2, \ldots, u_l\}$, if there exist $W \in V$ such that $\{(W, u_1), (W, u_2), \ldots, (W, u_l)\}$ belong to E, then $e = (H_e, v)$ is kept. Otherwise, $e = (H_e, v)$ is deleted from G. After all hyperedges are considered, we get a sub-hypergraph G' of G. Now an auxiliary graph $G_L = (V_L, E_L)$ based on G' is generated as follows. For each hyperedge $e = (H_e, v)$ in G', generate two super node h and w represent head set H_e and W respectively, then add directed edges (w, h) and (h, v). The influence probability of (w, h) and (h, v) are defined as $P^L_{(w,h)} = \prod_{i=1}^{i=l} P_{(W,u_i)}$ and $P^L_{(h,v)} = P_{(H_e,v)}$. Let V' contain all super nodes. Let the weight of super

nodes be 0, while weight of the other nodes be 1. Then, we define a weight function $f(\cdot)$ for node set.

$$f(v) = \begin{cases} 1, & v \in V \\ 0, & v \in V'. \end{cases}$$

The above process can be separated into two phrases. The first is to delete unsatisfied hyperedges while the second phrase is to generate super nodes and add new edges.

Fig. 5 shows how to generate directed graph for the lower bound. When we consider hyperedge (H, v) with probability P_4 in (a), there exists hyperedges (W, u_1), $(W, u_2), (W, u_3)$. Then hyperedge (H, v) can be kept. A hyperedge (H, v) in (c) will be deleted since u_1 and u_2 connect from different head sets W_1 and W_2. Graph (b) is a directed graph generated from (a). For head sets H and W, add two super nodes h and w. v will be activated by super node h with probability $P_4 = P_{(H,v)}$ and super node h will be activated by super node w with probability $P_{(w,h)} = P_{(W,u_1)}P_{(W,u_2)}P_{(W,u_3)} = P_1P_2P_3$. Also, super node w will be activated by node s with probability $P_{(s,w)} = P_{(s,w_1)}\, P_{(s,w_2)} = P_5P_6$.

Then auxiliary problem $G_L = (V \cup V', E_L, P^L, f)$ is to select k initially-influenced seed users in V to maximize the expected weighted number of eventually-influenced users, where E_L contains all original normal directed edges in G and new added edges.

$$\max \sigma_L(S) \tag{5}$$

$$s.t. |S| \leq k. \tag{6}$$

Where S is the initial seed set and $\sigma_L(S)$ is the expected weighted number of eventually-influenced nodes. The influence probability decreases in the hpyeredge deleting process and nodes merging process. We have the following theorem.

Theorem 4.2. *Given $G = (V, E, P)$, $\sigma_L(\cdot)$ is an lower bound of $\sigma(\cdot)$.*

Proof. We need to prove $\sigma_L(S) \leq \sigma(S)$ for any $S \in V$. According to the IC model, assume S'_t and S_t are the activated nodes in G_L and G respectively at step t. Since G^L contains super nodes, S'_t has two parts: nodes in V and nodes in V', where V is the original node set and V' is the super node set. Meanwhile, $\sigma_L(\cdot)$ is the weighted expected number of activated nodes. Then, we only need to prove $S'_t \cap V \subseteq S_t$ at each time step.

When $t = 0$, $S'_0 = S_0 = S \subseteq V$. First, we will prove $S'_1 \cap V \subseteq S_1$ after the first time step. For each inactivated node $v \in V$, we will prove that the total influence probability $P^L(v)$ in G_L is less than $P(v)$ in G. Assume v is the tail of l edges $(H_1, v), (H_2, v), \ldots, (H_l, v)$ in G. H_i will try to activate v with probability $P_{(H_i,v)}$ only if $H_i \subseteq S$. Then v can be activated with the probability $P(v) = 1 - \prod_{H_i \subseteq S}(1 - P_{(H_i,v)})$. On the other hand, some of these l edges will be deleted after the first phrase of formulating G'. Suppose $(H'_1, v), (H'_2, v), \ldots, (H'_q, v)$ are the $q \leq l$ edges which are kept in G'. For each hyperedge in these q edges, generate a new super node. Let $\{(H'_1, v), (H'_2, v), \ldots, (H'_q, v)\} = E_1 \cup E_2$, where $E_1 = \{(w_1, v), (w_2, v), \ldots, (w_h, v)\}$ is the normal edge set and E_2 is hyperedge set. Let $u \in U$ be the super node corresponding to hyperedge in E_2. Since $S \in V$ at the beginning, super nodes are inactivated. Then, $P^L(v) = 1 - \prod_{w_i \in S \cap \{w_1, w_2, \ldots, w_h\}}(1 - P_{(w_i,v)}) \leq 1 - \prod_{H_i \subseteq S}(1 - P_{(H_i,v)}) = P(v)$. Then, for each inactivated node $v \in V$, $P^L(v) \leq P(v)$ means $S'_1 \cap V \subseteq S_1$.

Second, suppose $S'_t \cap V \subseteq S_t$ at step t, we will prove $S'_{t+1} \cap V \subseteq S_{t+1}$ after one time step. For each inactivated node $v \in V$, we will prove that the total influence probability $P^L(v)$ in G_L is less than $P(v)$ in G. Assume v is the tail of l edges $(H_1, v), (H_2, v), \ldots, (H_l, v)$ in G. H_i will try to activate v with probability $P_{(H_i,v)}$ only if $H_i \subseteq S_t$. Then v can be activated with the probability $P(v) = 1 - \prod_{H_i \subseteq S_t}(1 - P_{(H_i,v)})$. On the other hand, some of these l edges will be deleted after the first phrase of formulating G'. Suppose $(H'_1, v), (H'_2, v), \ldots, (H'_q, v)$ are the $q \leq l$ edges which are kept in G'. For each hyperedge in these q edges, generate a new super node. Let $\{(H'_1, v), (H'_2, v), \ldots, (H'_q, v)\} = E_1 \cup E_2$, where $E_1 = \{(w_1, v), (w_2, v), \ldots, (w_h, v)\}$ is the normal edge set and E_2 is hyperedge set. Let $u \in U$ be the super node corresponding to hyperedge in E_2. Since $S'_t \in V \cup V'$, super nodes may be activated. For each activated super node, the corresponding hyperedge must be acitvated, that means all nodes in head set of this hyperedge must be activated since these nodes are activated by same node set in last time step. Then, $P^L(v) = 1 - \prod_{w_i \in S'_t \cap \{w_1, w_2, \ldots, w_h\}}(1 - P_{(w_i,v)}) \prod_{u \in S'_t \cap U}(1 - P_{(u,v)}) \leq 1 - \prod_{w_i \in S'_t \cap V}(1 - P_{(w_i,v)}) \prod_{H_i \subseteq S'_t \cap V}(1 - P_{(H_i,v)})$. Since $S'_t \cap V \subseteq S_t$, we have $1 - \prod_{w_i \in S'_t \cap V}(1 - P_{(w_i,v)}) \prod_{H_i \subseteq S'_t \cap V}(1 - P_{(H_i,v)}) \leq 1 - \prod_{w_i \in S_t}(1 - P_{(w_i,v)}) \prod_{H_i \subseteq S_t}(1 - P_{(H_i,v)}) \leq 1 - \prod_{H_i \subseteq S_t}(1 - P_{(H_i,v)}) = P(v)$. Then, for each inactivated node $v \in V$, $P^L(v) \leq P(v)$ means $S'_{t+1} \cap V \subseteq S_{t+1}$ after one time step. □

4.3 Properties of the Bounds

The above two auxiliary problems are NP-hard since they are normal SIMP and weighted SIMP respectively. $\sigma_L(\cdot)$ and $\sigma_U(\cdot)$ are monotone and submodular under IC model.

5 RIS Algorithm Method

We will extend Dynamic-Stop-and-Stare(D-SSA) [9] algorithm to solve general weighted SIMP. Then an randomized algorithm base on greedy strategy is designed for solving SIMPH. At the end, an sandwich approximation framework will be proposed for analyzing performance of our algorithms.

In order to solve the lower bound and upper bound, we define a general weighted SIMP. Let weighted directed graph $G = (V, C, E, P, f)$ denote a general weighted influence maximization problem with candidate seed set $C \subseteq V$ under the IC model, where P is the influence probability and f is weight function of node. Especially, E is the edge set in which only contains normal directed edges. Suppose S is the initial seed set. Let $\sigma'(S) = \sum_{v \text{ is activated}} f(v)$ be the expected weighted number of eventually-influenced nodes. Find k initially-influenced seed users in C to maximize $\sigma'(S)$. It is easy to see that $\sigma'(S)$ is monotone and submodular. Since G is random graph, sampling method is necessary to estimate $\sigma'(S)$.

5.1 (ϵ, δ)-approximation

We recall the (ϵ, δ)-approximation in [12] that will be used in our algorithm. ϵ is absolute error of estimation and $(1 - \delta)$ is confidence.

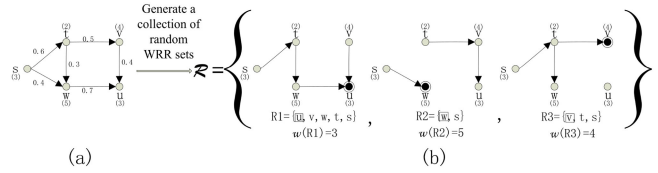


Fig. 6. An example for generating random WRR sets under IC model. R_1, R_2, R_3 with $w(R_1) = 3, w(R_2) = 5$, and $w(R_1) = 4$ are generated. (a) is the original weighted random graph. (b) contains three WRR sets up to three sample graphs.

Definition 5.1 ((ϵ, δ)-approximation). *Let $Z_1, Z_2, \ldots$ be independently and identically distributed samples according to Z in the interval [0,1] with mean μ_z and variance σ_Z^2. A Monte Carlo estimator of μ_z,*

$$\hat{\mu}_Z = \frac{1}{T}\sum_{i=1}^{T} Z_i, \tag{7}$$

is said to be an (ϵ, δ)-approximation of μ_Z if

$$Pr[(1-\epsilon)\mu_Z \le \hat{\mu}_Z \le (1+\epsilon)\mu_Z] \ge 1-\delta. \tag{8}$$

Define $\Upsilon = 4(e-2)\ln(2/\delta)/\epsilon^2$ and $\Upsilon_1 = 1 + (1+\epsilon)\Upsilon$, then the Stopping Rule Algorithm given in [12] has been proved to be (ϵ, δ)-approximation.

Lemma 5.1. *Let $Z_1, Z_2, \ldots$ be independently and identically distributed samples according to Z in the interval [0,1] with mean μ_z. Let $SumZ = \sum_{i=1}^{N} Z_i$, $\hat{\mu}_Z = \frac{SumZ}{N}$, $\Upsilon = 4(e-2)\ln(2/\delta)/\epsilon^2$ and $\Upsilon_1 = 1 + (1+\epsilon)\Upsilon$. If N is the number of samples when $SumZ \ge \Upsilon_1$, then $Pr[(1-\epsilon)\mu_Z \le \hat{\mu}_Z \le (1+\epsilon)\mu_Z] \ge 1-\delta$ and $\mathbb{E}[N] \le \Upsilon_1/\mu_Z$.*

5.2 RIS Sampling

First, we will introduce reverse influence set(RIS) sampling method [8]. Given a graph $G = (V, C, E, P, f)$, where $C \subseteq V$ is a candidate seed set. RIS captures the influence landscape of G through generating a set $\mathcal{R}$ of random *Weighted Reverse Reachable(WRR) sets*. Each WRR set R_j is a subset of V and constructed as follows,

Definition 5.2. *(Weighted Reverse Reachable set). Given $G = (V, C, E, P, f)$, a random WRR set R_j is generated from G by (1) selecting a random node $v \in V$; (2) generating a sample graph g from G; (3) returning R_j as the set of nodes that can reach v in g; and (4) $w(R_j) = f(v)$.*

For a seed set S, denote the coverage number of set S as $Cov_{\mathcal{R}}(S) = \sum_{R_j \in \mathcal{R}} \min\{|S \cap R_j|, 1\}$ and the coverage weight as $WCov_{\mathcal{R}}(S) = \sum_{R_j \in \mathcal{R}} w(R_j)\min\{|S \cap R_j|, 1\}$. $\sigma'(S)$ can be estimated by computing weighted coverage of set S. Fig. 6 shows an example of generating a collection of random WRR sets. Suppose seed set $S = \{t\}$, then $Cov_{\mathcal{R}}(S) = 2$ and $WCov_{\mathcal{R}}(S) = 7$.

Lemma 5.2. *Given $G = (V, C, E, P, f)$, a random WRR set R_j generated from G. For each seed set $S \subseteq C$, where $C \subseteq V$ is candidate seed set,*

$$\sigma'(S) = \sum_{v \in V} f(v) Pr[S\, covers\, R_j].$$

Proof.

$$\begin{aligned}\sigma'(S) &= \mathbb{E}[\sum_{\text{v is activated}} f(v)] \\ &= \sum_{v \in V} f(v) Pr[v \text{ is activated}] \\ &= \sum_{v \in V} f(v) Pr[S \text{ covers } R_j].\end{aligned}$$

□

Lemma 5.3. *The Greedy Weighted Max-Coverage returns an $(1-1/e)$-approximate seed set $\hat{S}_k$.*

Algorithm 1. Weighted Max-Coverage Procedure

Input: WRR sets ($\mathcal{R}$), k and weight function $w(\cdot)$.
Output: An $(1-1/e)$-approximation solution $\hat{S}_k$ and its estimated influence $\hat{\sigma'}(\hat{S}_k)$.

$\hat{S}_k = \emptyset$
for $i = 1$ to k **do**
  $\hat{v} \leftarrow \arg\max_{v \in C}(WCov_{\mathcal{R}}(\hat{S}_k \cup \{v\}) - WCov_{\mathcal{R}}(\hat{S}_k))$
  Add $\hat{v}$ to $\hat{S}_k$
end for
$\hat{\sigma'}(\hat{S}_k) = WCov_{\mathcal{R}}(\hat{S}_k) \cdot \sum_{v \in V} f(v) / \sum_{j=1}^{|\mathcal{R}|} w(R_j))$
return $< \hat{S}_k, \hat{\sigma'}(\hat{S}_k) >$.

In the sampling process, the most important thing is to determine the number of samples to satisfy the given estimation error. According to Dynamic-Stop-and-Stare algorithm in [9], we can prove the following D-SSA algorithm preserves the $(1-1/e-\epsilon)$-approximation factor.

Theorem 5.1. *Give ϵ, δ and a general weighted SIMP $G = (V, C, E, P, f)$, Algorithm 2 returns a $(1-1/e-\epsilon)$-approximation solution.*

Algorithm 2. D-SSA Algorithm for General Weighted SIMP

Input: Graph $G = (V, C, E, P, f)$, $n = |V|$, $0 \le \epsilon, \delta \le 1$ and k.
Output: An $(1-1/e-\epsilon)$-approximation solution $\hat{S}_k$.

$\Gamma \leftarrow 4(e-2)(1+\epsilon)^2 \ln(2/\delta)(1/\epsilon^2)$
$\mathcal{R} \leftarrow$ generate Γ random RR sets by RIS
$< \hat{S}_k, \hat{\sigma'}(\hat{S}_k) > \leftarrow$ Weighted Max-Coverage($\mathcal{R}, k, f(\cdot)$)
while $|\mathcal{R}| \ge (8+2\epsilon)n \cdot \frac{\ln(\frac{2}{\delta}) + \ln C_n^k}{\epsilon^2}$ **do**
  $\mathcal{R}' \leftarrow$ generate Γ random RR sets by RIS
  $\sigma'_c(\hat{S}_k) \leftarrow WCov_{\mathcal{R}'}(\hat{S}_k) \cdot \sum_{v \in V} f(v) / \sum_{j=1}^{|\mathcal{R}'|} w(R_j)$
  $\epsilon_1 \leftarrow \hat{\sigma'}(\hat{S}_k)/\sigma'_c(\hat{S}_k) - 1$
  if $(\epsilon_1 \le \epsilon)$ **then**
    $\epsilon_2 \leftarrow \frac{\epsilon - \epsilon_1}{2(1+\epsilon_1)}, \epsilon_3 \leftarrow \frac{\epsilon - \epsilon_1}{2(1-1/e)}$
    $\delta_1 \leftarrow e^{-\frac{Cov_{\mathcal{R}}(\hat{S}_k)\epsilon_3^2}{2c(1+\epsilon_1)(1+\epsilon_2)}}$
    $\delta_2 \leftarrow e^{-\frac{(Cov_{\mathcal{R}'}(\hat{S}_k)-1)\epsilon_2^2}{2c(1+\epsilon_2)}}$
    if $\delta_1 + \delta_2 \le \delta$ **then**
      return $\hat{S}_k$
    end if
  end if
  $\mathcal{R} \leftarrow \mathcal{R} \cup \mathcal{R}'$
  $< \hat{S}_k, \hat{\sigma'}(\hat{S}_k) > \leftarrow$ Weighted Max-Coverage($\mathcal{R}, k, f(\cdot)$)
end while
return $\hat{S}_k$

5.3 Greedy Strategy for SIMPH

In this section, (ϵ, δ)-approximation calculation procedure of $\sigma(S)$ on a random hypergraph for a given seed set S is proposed first. Then, an algorithm based on greedy strategy for SIMPH will be presented.

5.3.1 Influence Estimation

Given a directed hypergraph $G = (V, E, P)$ with n nodes, $\sigma(S)$ is the expected number of eventually-influenced nodes for seed set S. Suppose $g = (V, E')$ is a sample graph of G, let $\sigma_g(S)$ denote the number of eventually-influenced nodes. Then $\frac{\sigma_g(S)}{n}$ is random variable distributed in interval $[0, 1]$. The (ϵ, δ)-approximation calculation procedure of $\sigma(S)$ for a seed set S is as follows.

Algorithm 3. APP-Calculation Procedure

Input: A directed hypergraph $G = (V, E, P)$, $n = |V|$, $0 \leq \epsilon, \delta \leq 1$, seed set S.
Output: $\sigma_c(S)$ such that $\sigma_c(S) \leq (1+\epsilon)\sigma(S)$ with at least $(1-\delta)$-probability

$\Upsilon_1 = 1 + 4(1+\epsilon)(e-2)\ln(2/\delta)/\epsilon^2$
$SumZ = 0$
$N = 0$
while $SumZ \leq \Upsilon_1$ **do**
  $g \leftarrow$ generate sample graph of G
  $N = N + 1, S_1 = S, S_2 = S$
  while $S_2 \neq \emptyset$ **do**
    $S_1 = S_1 \cup S_2$
    $S_2 = \emptyset$
    for each hyperedge $e = (H_e, v) \in E$ in g and v is inactive **do**
      if $H_e \subseteq S_1$ **then**
        Add v to S_2
      end if
    end for
  end while
  $SumZ = SumZ + \frac{|S_1|}{n}$
end while
return $\sigma_c(S) = n \cdot \frac{SumZ}{N}$

According to Lemma 5.1, we need to generate N graphs that satisfy the stopping rule $\sum_{i=1}^{N} \frac{\sigma_{g_i}(S)}{n} \geq 1 + 4(1+\epsilon)(e-2)\ln(2/\delta)/\epsilon^2$. From Lemma 5.1, we obtain a direct corollary as stated below,

Corollary 5.1. *The APP-Calculation procedure returns an estimate $\sigma_c(S)$ of $\sigma(S)$ such that*

$$Pr[(1-\epsilon)\sigma(S) \leq \sigma_c(S) \leq (1+\epsilon)\sigma(S)] \geq 1 - \delta. \quad (9)$$

5.3.2 Greedy Algorithm

The nodes in the head set of A hyperedge will try to active the tail node only when they are all active. The reverse technique in RIS sampling is unsuitable. Then, we design a greedy algorithm, as shown in Algorithm 4. Starting with an empty seed set, the greedy strategy iteratively adds a node that maximizes the marginal gain of $\sigma(S)$, until k nodes are selected.

Algorithm 4. Greedy Strategy for SIMPH

Input: A directed hypergraph $G = (V, E, P)$, k.
Output: A set of seed nodes, S.

$S = \emptyset$
for $i = 1$ to k **do**
  $v \leftarrow \arg\max_{v \in V}$(APP-Calculation$(G, S \cup \{v\})$−APP-Calculation$(G, S)$)
  Add v to S
end for
return S

5.4 Sandwich Approximation Framework

For SIMPH, we have provide a lower bound and an upper bound for $\sigma(\cdot)$ in Section 4. Then, the sandwich approximation framework is shown in Algorithm 5.

Algorithm 5. Sandwich Approximation Framework

Input: A directed hypergraph $G = (V, E, P)$, k, ϵ, δ.
Output: A set of seed nodes, S.

Let S_L be the output seed set of solving the auxiliary problem $G_L = (V \cup V', E_L, P^L)$ for lower bound by D-SSA Algorithm (Algorithm 2).
Let S_U be the output seed set of solving the auxiliary problem $G_U = (V, E_U, P^U)$ for upper bound by D-SSA Algorithm (Algorithm 2).
Let S_A be the output seed set of solving $G = (V, E, P)$ by Greedy Strategy for SIMPH(Algorithm 4).
$S =$arg $\max_{S_0 \in \{S_L, S_U, S_A\}}$APP-Calculation$(G, S_0)$
return S

For sandwich approximation framework, we can get the following result.

Theorem 5.2. *Let S be the seed set returned by Algorithm 5, then we have*

$$\sigma(S) \geq \max\{\frac{\sigma(S_U)}{\sigma_U(S_U)}, \frac{\sigma_L(S_L^*)}{\sigma(S^*)}\}\frac{1-\epsilon}{1+\epsilon}(1 - \frac{1}{e} - \epsilon)\sigma(S^*). \quad (10)$$

Where S_L^* is the optimal solution to maximize the lower bound problem and S^* is the optimal solution of SIMPH.

Proof. Let S_U^* be the optimal solution to maximize the upper bound IM problem. Then, we have

$$\sigma(S_U) = \frac{\sigma(S_U)}{\sigma_U(S_U)}\sigma_U(S_U) \geq \frac{\sigma(S_U)}{\sigma_U(S_U)}(1 - \tfrac{1}{e} - \epsilon)\sigma_U(S_U^*)$$
$$\geq \frac{\sigma(S_U)}{\sigma_U(S_U)}(1 - \tfrac{1}{e} - \epsilon)\sigma_U(S^*) \geq \frac{\sigma(S_U)}{\sigma_U(S_U)}(1 - \tfrac{1}{e} - \epsilon)\sigma(S^*)$$

and

$$\sigma(S_L) \geq \sigma_L(S_L) \geq (1 - \frac{1}{e} - \epsilon)\sigma_L(S_L^*) \geq \frac{\sigma_L(S_L^*)}{\sigma(S^*)}(1 - \frac{1}{e} - \epsilon)\sigma(S^*).$$

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

$$\sigma(S_{max}) \geq \max\left\{\frac{\sigma(S_U)}{\sigma_U(S_U)}, \frac{\sigma_L(S_L^*)}{\sigma(S^*)}\right\}\left(1 - \frac{1}{e} - \epsilon\right)\sigma(S^*).$$

Since $\forall S_0 \in \{S_L, S_U, S_A\}$, $(1-\epsilon)\sigma(S_0) \leq \sigma_c(S_0) \leq (1+\epsilon)\sigma(S_0)$, we have

$$(1+\epsilon)\sigma(S) \geq \sigma_c(S) \geq \sigma_c(S_{max}) \geq (1-\epsilon)\sigma(S_{max}).$$

It follows that

$$\sigma(S) \geq \tfrac{1-\epsilon}{1+\epsilon}\sigma(S_{max}) \geq \max\{\tfrac{\sigma(S_U)}{\sigma_U(S_U)}, \tfrac{\sigma_L(S_L^*)}{\sigma(S^*)}\}\tfrac{1-\epsilon}{1+\epsilon}(1-\tfrac{1}{e}-\epsilon)\sigma(S^*) \quad \square$$

According to Theorem 5.2, the difference between $\sigma(S^*)$ and $\sigma_L(S_L^*)$ has great influence on the performance of Algorithm 5. Iyer and Bilmes [13] studied the minimization problem of the difference between submodular function. While the difference between $\sigma(S^*)$ and $\sigma_L(S_L^*)$ may be bounded, we have the following result.

Theorem 5.3. *Let S_L^* be the optimal solution to maximize the lower bound problem and S^* is the optimal solution of SIMPH, then we have*

$$\sigma(S^*) - \sigma_L(S_L^*) \leq \max_{S,|S|=k}(\sigma_U(S) - \sigma_L(S)). \tag{11}$$

Proof.

$$\sigma(S^*) - \sigma_L(S_L^*) \leq \sigma(S^*) - \sigma_L(S^*) \leq \sigma_U(S^*) - \sigma_L(S^*) \leq \max_{S||S|=k}(\sigma_U(S) - \sigma_L(S)).$$

$\square$

Additionally, the structure of graph will have significantly impact on effectiveness of the lower bound. According to the lower bound formulation, hyperedges will be deleted if nodes in their head set do not have a same head set. An extreme case is all hyperedge are deleted, which is the worst case of lower bound. In reality, this case may not appear which could be seen in the Experiments. We also do some small size experiments to evaluate $\sigma_L(S_L^*)$ with $\sigma(S^*)$.

6 Experiments

6.1 Statistics and Information of Datasets

Our experiments are based on the 3 datasets from Forum on toreopsahl.com which is an online forum network. Each dataset has both one mode and two mode data which are preprocessed to give the edges (both simple and hyperedges) of the graph on which the experiments are performed. The first dataset is the Facebook-like Forum Network [14], the second dataset is Newmans scientific collaboration network [15] and the third dataset is Norwegian Interlocking Directorate [16]. The statistics of the data (after processing to graphs to form hyperedges) are represented in Table 2. The first dataset logs information of user interaction on different topics discussed in the forum. It contains three columns namely a time stamp, a user and a topic that the user had liked. This data is preprocessed to build the hypergraph with hyperedges (crowd influence) in the form of (H_e, v) where $H_e \geqslant 1$ and v is the tail node. The second dataset used for our experiments is in the Newmans scientific collaboration network. This is a network representing the co-authorship network. The dataset had two columns namely author and paper ID and the corresponding hyperedges are generated based on the relationship between co-authors and the paper that they have written. The third dataset is in the Norwegian Interlocking Directorate Network. This data represents the interlinking relationship among directors and among 384 public limited companies on the website of the Norwegian business site. The dataset contains two columns namely company and director and the interlocking connection among them is derived to form the hyperedges of our graph. All the programs are written in Python 3.6.3 and run on a Linux server with 16 CPUs and 251 GB RAM.

TABLE 2
Data Statistics

	Normal Directed Edges	Hyperedges	Nodes
Dataset1	142,760	479	897
Dataset2	95,188	3,668	16,264
Dataset3	7,710	4,977	2,045

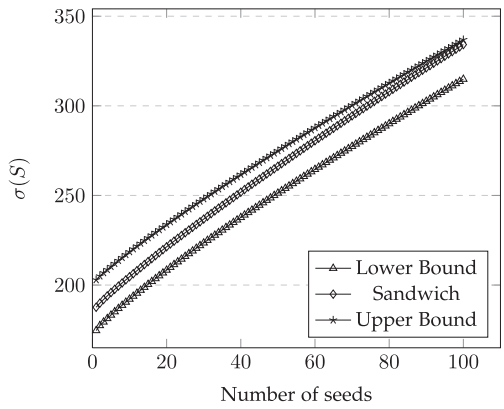
6.2 Pre-Processing of Data and Crowd Influence Detection

6.2.1 Dataset 1

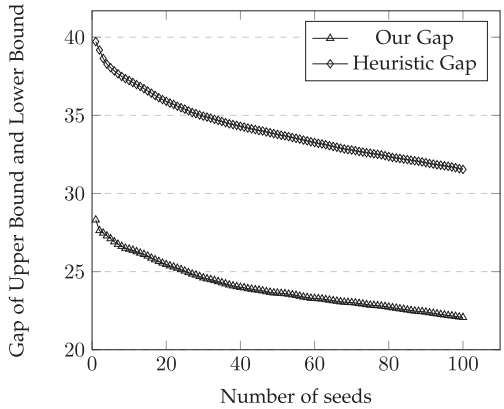
The first dataset is the Facebook-like Forum Network. It has both one mode and two mode data. The one mode data contains relationship among Person A and Person B and the number of messages that they have shared. This one mode dataset is used to form direct edges among people (nodes) in the graph and the number of messages they shared are assigned as the weight of the edge between A and B. The two mode network represents 899 users and 522 topics and describes which user had liked which topic sorted by time. The hyperedges are built by analyzing the two mode data. The concept to identify the influence is that if Person A liked a topic X at time T1 and Person B liked the same topic X at time T2 and Person C liked the same topic X at time T3 and if T1$\leq$ T2 $<$ T3 then we can establish an edge going from (A,B) to C by assuming that A and B have influenced C in liking the topic X. The edge weight for this hyperedge is assigned by calculating how many topics the influencing entities are interested in. The dataset is sorted according to times tamp and a sorted list was made of all the people who have liked/read a topic and based on the time stamp, the hyperedges of the graph are built and the corresponding weights are assigned. After pre-processing of the data, the final graph have 897 nodes, normal edge count of 142,760 and hyperedge count of 479. The graphs below show the experimental results for dataset 1.

6.2.2 Dataset 2

The second dataset is the Newmans scientific collaboration network. The one mode data in this dataset records the relationship between Author A and Author B and the total number of papers that they have written together. The directed edges are built from the one mode data and the total number of papers that the two authors have written is assigned as the weight of the the edge between them. For building the hyperedges of the graph, the two mode data is processed. The two mode data gives the information about the co-authors of a given paper. The head set for the hyperedge is derived by analysing the total number of papers that each pair of co-authors have written. For example, if A,B and C are three authors who have written paper 1 together, we analyze how many papers (A,B), (B,C) and (A,C) have written. Whichever pair has written the most papers is assumed to influence the the other. Let us assume the number of papers for (A,B) is 5, for (B,C) is 2 and (A,C) is 3, then we can establish a hyperedge (A,B) to C by

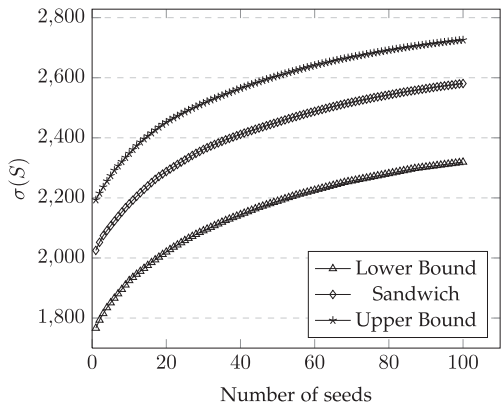


(a) Sandwich Framework

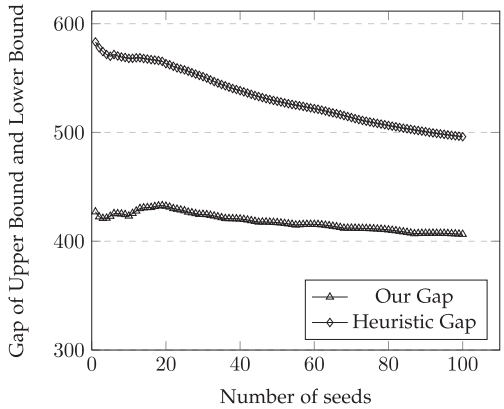


(b) Gap Comparison of Upper Bound and Lower Bound

Fig. 7. Experimental results for Dataset 1.



(a) Sandwich Framework



(b) Gap Comparison of Upper Bound and Lower Bound

Fig. 8. Experimental results for Dataset 2.

assuming that as (A,B) had the maximum number of papers, they must have influenced author C to collaborate with them. The weight associated with this hyperedge would the the total number of papers that the influencing entity have written together. After pre-processing the data, the final graph have 16,264 nodes, normal edge count of 95188 and hyperedge count is 3668. The graphs below show the experimental results for dataset 2.

6.2.3 Dataset 3

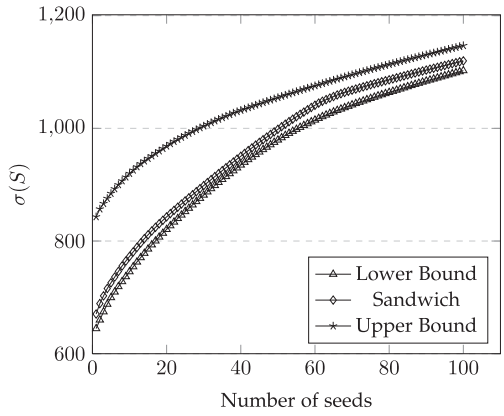
The third dataset is the Norwegian Interlocking Directorate network. The one mode data in this dataset depicts the relationship among directors in a company. The relation among directors A and B are used to build a simple edge among them. For the hyperedges, the two mode data is processed. The two mode data gives the list of directors in a company. For establishing an influence relation, we analyze the list of directors in companies through months over three years. For example, lets assume company C1 has a list L1 = {A,B, C} of directors in January of 2008, in April of 2008, the list changes to L2 = {B,C,D,E}, in July of 2008 the list changes to L3 = {B,C,F} and in October of 2008 the list changes to L4 = {B,F}. By analyzing the lists L1, L2, L3 and L4 we can see that by April of 2008 director A had left the company, by July of 2008 directors D and E had left the company and by October of 2008 director C had left the company. Thus we can assume that directors A and D had influenced C to leave or directors A and E had influenced C to leave. Thus we can establish the hyperedges (A,D) to C and (A,E) to C. The weight assigned to these hyperedges is the maximum time difference of the entity leaving the company. For example, A left in April and C left in October, then the difference is 10-4 = 6. After pre-processing the data, the final graph had 2045 nodes, normal edge count is 7710 and hyperedge count is 4,977.

6.3 Performance and Comparison

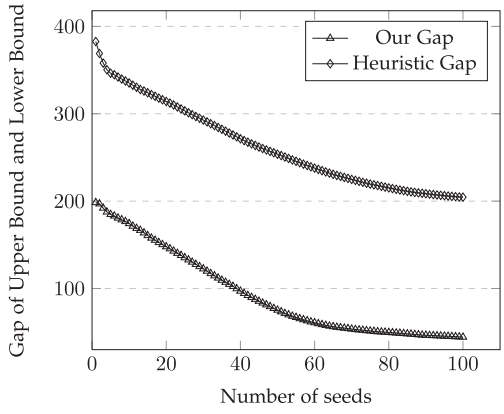
Two experiments are performed for each dataset after the graphs for each dataset is built using the concepts in Section 6.2. The first experiment is performed to obtain $\sigma(S)$ values for a given seed set value k (ranging from 0 to 100) for upper bound, the sandwich framework and lower bound. The second experiment is performed to obtain the difference between upper bound and lower bound for a value of k ranging between 0 to 100 and compared to a heuristic gap. All the experiments are plotted on graphs and shown in Figs. 7, 8 and 9.

6.4 Experimental Results

From the graphs in Figs. 7, 8 and 9 it is observed that in the first experiment the sandwich framework gave $\sigma(S)$ values lying in between the $\sigma(S)$ values for the upper bound and lower bound for all the three datasets and the $\sigma(S)$ values increased with increasing values of seed set k. In the second experiment



(a) Sandwich Framework



(b) Gap Comparison of Upper Bound and Lower Bound

Fig. 9. Experimental results for Dataset 3.

it is observed that our gap is significantly less than the heuristic gap for all the three datasets and also the gap decreased with the increase in the value of seed set k. While the lower bound in heuristic gap is the expected number of influenced nodes without considering any crowd influence.

6.5 Experiments on Small Hypergraphs

In this section, we do experiments on small hypergraphs. We generate 10 random hypergraphs of 20 nodes and randomly generate its normal edges and hyperedges. We use these graphs to compare the results of Algorithm 5 with the optimal solution and also to compare with the value of lower bound problems. We use the enumerate algorithm to obtain the optimal solution. The number of seeds k values used in the experiment are $\{1, 2, 3\}$ as higher k values would be time consuming. Two results can be observed from the experiments on these small hypergraphs. First result observed is that the output for Algorithm 5 and the optimal solution are same and the second result observed is that the lower bound output and the output of Algorithm 5 have a very small gap. Table 3 shows the average values of these 10 experiments.

TABLE 3
Experiment Results on Small Hypergraphs

Number of Seeds	1	2	3
σ for Algorithm 5	18.3849	19.3415	19.6723
σ for Optimal Solution	18.3849	19.3415	19.6723
σ_L for lower bound	16.8842	18.5328	19.3617

7 Conclusion

In this paper, we modeled the crowd influence in information diffusion process by using a hyperedge. Social Influence Maximization Problem in Hypergraph was formulated to select initially-influenced seed users under Independent Cascade model to maximize the expected number of eventually-influenced users. We showed SIMPH is NP-hard and the objective function was neither submodular nor supermodular. We developed a lower bound and an upper bound so that the Sandwich framework can applied. We presented a D-SSA algorithm to solve the lower bound and upper bound which were general weighted social influence maximization problem. Then, a greedy strategy based on influence increment maximization with randomized algorithm for estimation of the objective function in SIMPH was presented. Finally, we verified our algorithm on real world data sets. The results showed crowd influence played an important role in the information diffusion process. For future research, we are looking for an efficient method to solve nonsubmodular problems, such as SIMPH and also to be able to incorporate the effect of crowd influence to formulate new models of social networks.

Acknowledgments

This work was supported in part by the US National Science Foundation (NSF) under Award no. 1747818, China US National Science Foundation (CNSF) under Grant no. 61472272, 91324012, 91024031, 11771403, and Zhejiang Provincial Natural Science Foundation of China under Grant no. LY17A010025.

References

[1] H. Nguyen and R. Zheng, "On budgeted influence maximization in social networks," *IEEE J. Sel. Areas Commun.*, vol. 31, no. 6, pp. 1084–1094, Jun. 2013.
[2] M. Edelson, T. Sharot, R. J. Dolan, and Y. Dudai, "Following the crowd: Brain substrates of long-term memory conformity," *Sci.*, vol. 333, no. 6038, pp. 108–111, 2011.
[3] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2003, pp. 137–146.
[4] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance, "Cost-effective outbreak detection in networks," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2007, pp. 420–429.
[5] Y. Wang, G. Cong, G. Song, and K. Xie, "Community-based greedy algorithm for mining top-k influential nodes in mobile social networks," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2010, pp. 1039–1048.
[6] Y. Tang, X. Xiao, and Y. Shi, "Influence maximization: Near-optimal time complexity meets practical efficiency," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2014, pp. 75–86.
[7] Y. Tang, Y. Shi, and X. Xiao, "Influence maximization in near-linear time: A martingale approach," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2015, pp. 1539–1554.
[8] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier, "Maximizing social influence in nearly optimal time," in *Proc. 25th Annu. ACM-SIAM Symp. Discr. Algorithms*, 2014, pp. 946–957.
[9] H. T. Nguyen, M. T. Thai, and T. N. Dinh, "Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks," in *Proc. Int. Conf. Manage. Data*, 2016, pp. 695–710.

[10] H.-J. Hung, H.-H. Shuai, D.-N. Yang, L.-H. Huang, W.-C. Lee, J. Pei, and M.-S. Chen, "When social influence meets item inference," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 915–924.
[11] W. Lu, W. Chen, and L. V. Lakshmanan, "From competition to complementarity: Comparative influence diffusion and maximization," *Proc. VLDB Endowment*, vol. 9, no. 2, pp. 60–71, 2015.
[12] P. Dagum, R. Karp, M. Luby, and S. Ross, "An optimal algorithm for monte carlo estimation," *SIAM J. Comput.*, vol. 29, no. 5, pp. 1484–1496, 2000.
[13] R. Iyer and J. Bilmes, "Algorithms for approximate minimization of the difference between submodular functions, with applications," in *Proc. 28th Conf. Uncertainty Artif. Intell.*, 2012, pp. 407–412.
[14] T. Opsahl, "Triadic closure in two-mode networks: Redefining the global and local clustering coefficients," *Social Netw.*, vol. 35, no. 2, pp. 159–167, 2013.
[15] M. E. Newman, "The structure of scientific collaboration networks," *Proc. Nat. Acad. Sci. United States America*, vol. 98, no. 2, pp. 404–409, 2001.
[16] C. Seierstad and T. Opsahl, "For the few not the many? the effects of affirmative action on presence, prominence, and social capital of women directors in norway," *Scandinavian J. Manage.*, vol. 27, no. 1, pp. 44–54, 2011.

Jianming Zhu received the BS and MS degrees in mathematics from Shandong University, in 2001 and 2004, respectively, and the PhD degree in operations research from the Chinese Academy of Sciences, in 2007. He is an associate professor in the School of Engineering Science, University of Chinese Academy of Sciences and a visiting scientist in the Department of Computer Science, University of Texas at Dallas. His research focuses on algorithm design and analysis for optimization problems in data science, wireless networks, and management science.

Junlei Zhu is currently working toward the PhD degree in the College of Mathematics, Physics and Information Engineering, Zhejiang Normal University, Jinhua, Zhejiang, China. She is a lecturer in the College of Mathematics, Physics and Information Engineering, Jiaxing University, Jiaxing, Zhejiang, China. Her research interests include graph theory, algorithm, and social networks.

Smita Ghosh received the bachelor's degree in computer science and engineering from the West Bengal University of Technology, Kolkata, West Bengal, India, in May 2015, and the master's degree in computer science from the University of Texas at Dallas and graduated, in May 2017. She is currently working toward the full time computer science PhD degree working with Dr. Weili Wu, in the Department of Computer Science, University of Texas at Dallas. Her research interests are analysis of social networks and incorporating the big data analysis of social network with machine learning and deep learning.

Weili Wu (M'00) received the MS and PhD degrees from the Department of Computer Science, University of Minnesota, Minneapolis, Minnesota, in 1998 and 2002, respectively. She is currently a full professor with the Department of Computer Science, University of Texas at Dallas, Dallas, Texas. Her research mainly deals in the general research area of data communication and data management. Her research focuses on the design and analysis of algorithms for optimization problems that occur in wireless networking environments and various database systems. She is a member of the IEEE.

Jing Yuan received the BS degree in computer science from Nanjing University. She has been working toward the PhD degree since 2015 at the University of Texas at Dallas. Her research interests span social computing, ecommerce, combinatorial optimization, and algorithms. She is a student member of IEEE and ACM.