

When Social Influence Meets Item Inference

Hui-Ju Hung*, Hong-Han Shuai†, De-Nian Yang†, Liang-Hao Huang†,
Wang-Chien Lee*, Jian Pei§, Ming-Syan Chen‡

*The Pennsylvania State University, State College, Pennsylvania, USA

†Academia Sinica, Taipei, Taiwan

§Simon Fraser University, Burnaby, Canada

‡National Taiwan University, Taipei, Taiwan

ABSTRACT

Research issues and data mining techniques for product recommendation and viral marketing have been widely studied. Existing works on seed selection in social networks do not take into account the effect of product recommendations in e-commerce stores. In this paper, we investigate the seed selection problem for viral marketing that considers both effects of social influence and item inference (for product recommendation). We develop a new model, *Social Item Graph (SIG)*, that captures both effects in form of hyperedges. Accordingly, we formulate a seed selection problem, called *Social Item Maximization Problem (SIMP)*, and prove the hardness of SIMP. We design an efficient algorithm with performance guarantee, called Hyperedge-Aware Greedy (HAG), for SIMP and develop a new index structure, called SIG-index, to accelerate the computation of diffusion process in HAG. Moreover, to construct realistic SIG models for SIMP, we develop a statistical inference based framework to learn the weights of hyperedges from data. Finally, we perform a comprehensive evaluation on our proposals with various baselines. Experimental result validates our ideas and demonstrates the effectiveness and efficiency of the proposed model and algorithms over baselines.

1. INTRODUCTION

The ripple effect of social influence [4] has been explored for viral marketing via online social networks. Indeed, studies show that customers tend to receive product information from friends better than advertisements on traditional media [19]. To explore the potential impact of social influence, many research studies on *seed selection*, i.e., selecting a given number of influential customers to maximize the spread of social recommendation for a product, have been reported [7, 17].¹ However, these works do not take into account the effect of product recommendations in online e-commerce stores. We argue that when a customer buys an item due to the social influence (e.g., via Facebook or Pinterest), there is a potential side effect due to the *item in-*

¹All the top 5 online retailers, including Amazon, Staples, Apple, Walmart, and Dell, are equipped with sophisticated recommendation engines. They also support viral marketing by allowing users to share favorite products in Facebook.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

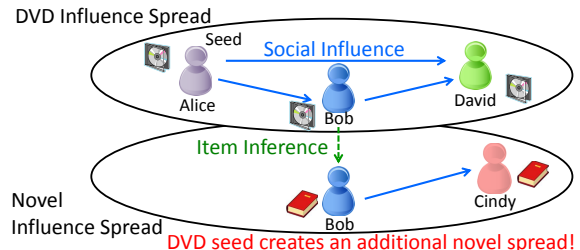


Figure 1: A motivating example

ference recommendations from stores.² For example, when Alice buys a DVD of “Star War” due to the recommendation from friends, she may also pick up the original novel of the movie due to an in-store recommendation, which may in turn trigger additional purchases of the novel among her friends. To the best of our knowledge, this additional spread introduced by the item inference recommendations has not been considered in existing research on viral marketing.

Figure 1 illustrates the above joint effects in a toy example with two products and four customers, where a dash arrow represents the association rule behind the item inference recommendation, and a solid arrow denotes the social influence between two friends upon a product. In the two separate planes corresponding to DVD and novel, social influence is expected to take effect on promoting interests in (and potential purchases of) the DVD and novel, respectively. Meanwhile, the item inference recommendation by the e-commerce store is expected to trigger sales of additional items. Note that the association rules behind item inference are derived without considering the ripple effect of social influence. In the example, when Bob buys the DVD, he may also buy the novel due to the item inference recommendation. Moreover, he may influence Cindy to purchase novel. However, the association rules behind item inference are derived without considering the ripple effect of social influence. On the other hand, to promote the movie DVD, Alice may be selected as a seed for a viral marketing campaign, hoping to spread her influence to Bob and David to trigger additional purchases of the DVD. Actually, due to the effect of item inference recommendation, having Alice as a seed may additionally trigger purchases of the novel by Bob and Cindy. This is a factor that existing seed selection algorithms for viral marketing do not account for.

We argue that to select seeds for maximizing the spread of product information to a customer base (or maximizing the sale revenue of products) in a viral marketing campaign, both effects of item inference and social influence need to

²In this paper, we refer product/item recommendation based on associations among items inferred from purchase transactions as item inference recommendation.

be considered. To incorporate both effects, we propose a new model, called *Social Item Graph (SIG)* in form of hyperedges, for capturing “purchase actions” of customers on products and their potential influence to trigger other purchase actions. Different from the conventional approaches [7, 17] that use links between customers to model social relationship (for viral marketing) and links between items to capture the association (for item inference recommendation), SIG represents a *purchase action* as a node (denoted by a tuple of a customer and an item), while using hyperedges among nodes to capture the influence spread process used to predict customers’ future purchases. Unlike the previous influence propagation models [7, 17] consisting of only one kind of edges connecting two customers (in social influence), the hyperedges in our model span across tuples of different customers and items, capturing both effects of social influence and item inference.

Based on SIG, we formulate the *Social Item Maximization Problem (SIMP)* to find a seed set, which consists of selected products along with targeted customers, to maximize the total adoptions of products by customers. Note that SIMP takes multiple products into consideration and targets on maximizing the number of products purchased by customers.³ SIMP is a very challenging problem, which does not have the submodularity property. We prove that SIMP cannot be approximated within n^c with any $c < 1$, where n is the number of nodes in SIMP, i.e., SIMP is extremely difficult to approximate with a small ratio because the best approximation ratio is almost n .⁴

To tackle SIMP, two challenges arise: 1) numerous combinations of possible seed nodes, and 2) expensive on-line computation of influence diffusion upon hyperedges. To address the first issue, we first introduce the *Hyperedge-Aware Greedy (HAG)* algorithm, based on a unique property of hyperedges, i.e., a hyperedge requires all its source nodes to be activated in order to trigger the purchase action in its destination node. HAG selects multiple seeds in each seed selection iteration to further activate more nodes via hyperedges.⁵ To address the second issue, we exploit the structure of Frequent Pattern Tree (FP-tree) to develop *SIG-index* as an compact representation of SIG in order to accelerate the computation of activation probabilities of nodes in online diffusion.

Moreover, to construct realistic SIG models for SIMP, we also develop a statistical inference based framework to learn the weights of hyperedges from logs of purchase actions. Identifying the hyperedges and estimating the corresponding weights are major challenges for constructing of a SIG due to data sparsity and unobservable activations. To address these issues, we propose a novel framework that employs smoothed expectation and maximization algorithm (EMS) [21], to identify hyperedges and estimate their values by kernel smoothing.

Our contributions of this paper are summarized as follows.

- We observe the deficiencies in existing techniques for item inference recommendation and seed selection and propose the *Social Item Graph (SIG)* that captures both effects of social influence and item inference in prediction of potential purchase actions.

³SIMP can be extended to a weighted version with different profits from each product. In this paper, we focus on maximizing the total sales.

⁴While there is no good solution quality guarantee for the worst case scenario, we empirically show that the algorithm we developed achieves total adoptions on average comparable to optimal results.

⁵A hyperedge requires all its source nodes to be activated to diffuse its influence to its destination node.

- Based on SIG, we formulate a new problem, called *Social Item Maximization Problem (SIMP)*, to select the seed nodes for viral marketing that effectively facilitates the recommendations from both friends and stores simultaneously. In addition, we analyze the hardness of SIMP.
- We design an efficient algorithm with performance guarantee, called Hyperedge-Aware Greedy (HAG), and develop a new index structure, called SIG-index, to accelerate the computation of diffusion process in HAG.
- To construct realistic SIG models for SIMP, we develop a statistical inference based framework to learn the weights of hyperedges from data.
- We conduct a comprehensive evaluation on our proposals with various baselines. Experimental result validates our ideas and demonstrates the effectiveness and efficiency of the proposed model and algorithms over baselines.

The rest of this paper is organized as follows. Section 2 reviews the related work. Section 3 details the SIG model and its influence diffusion process. Section 4 formulates SIMP and designs new algorithms to efficiently solve the problem. Section 5 describes our approach to construct the SIG. Section 6 reports our experiment results and Section 7 concludes the paper.

2. RELATED WORK

To discover the associations among purchased items, frequent pattern mining algorithms find items which frequently appear together in transactions [3]. Some variants, such as closed frequent patterns mining [20], maximal frequent pattern mining [16], have been studied. However, those existing works, focusing on unveiling the common shopping behaviors of individuals, disregard the social influence between customers [26]. On the other hand, it has been pointed out that items recommended by item inference may have been introduced to users by social diffusion [25]. In this work, we develop a new model and a learning framework that consider both the social influence and item inference factors jointly to derive the association among purchase actions of customers. In addition, we focus on seed selection for prevalent viral marketing by incorporating the effect of item inference.

With a great potential in business applications, social influence diffusion in social networks has attracted extensive interests recently [7, 17]. Learning algorithms for estimating the social influence strength between social customers have been developed [10, 18]. Based on models of social influence diffusion, identifying the most influential customers (seed selection) is a widely studied problem [7, 17]. Precisely, those studies aim to find the best k initial seed customers to target on in order to maximize the population of potential customers who may adopt the new product. This seed selection problem has been proved as NP-hard [17]. Based on two influence diffusion models, Independent Cascade (IC) and Linear Threshold (LT), Kempe et al. propose a $1 - 1/e$ approximation greedy algorithm by exploring the submodularity property under IC and LT [17]. Some follow-up studies focus on improving the efficiency of the greedy algorithm using various spread estimation methods, e.g., MIA[7] and TIM+[24]. However, without considering the existence of item inference, those algorithms are not applicable to SIMP. Besides the IC and LT model, Markov random field has been used to model social influence and calculate expected profits from viral marketing [9]. Recently, Tang et al. proposed a Markov model based on “confluence”, which estimates the total influence by combining different sources

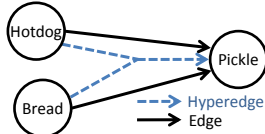


Figure 2: A hyperedge example

of conformity [23]. However, these studies only consider the diffusion of a *single* item in business applications. Instead, we incorporate item inference in spread maximization to estimate the influence more accurately.

3. SOCIAL ITEM GRAPH MODEL

Here we first present the social item graph model and then introduce the diffusion process in the proposed model.

3.1 Social Item Graph

We aim to model user purchases and potential activations of new purchase actions from some prior. We first define the notions of the social network and purchase actions.

DEFINITION 1. A social network is denoted by a directed graph $G = (V, E)$ where V contains all the nodes and E contains all the directed edges in the graph. Accordingly, a social network is also referred to as a social graph.

DEFINITION 2. Given a list of commodity items I and a set of customers V , a purchase action (or purchase for short), denoted by (v, i) where $v \in V$ is a customer, and $i \in I$ is an item, refers to the purchase of item i by customer v .

DEFINITION 3. An purchase log is a database consisting of all the purchase actions in a given period of time.

Association-rule mining (called item inference in this paper) has been widely exploited to discover correlations between purchases in transactions. For example, the rule $\{\text{hotdog}, \text{bread}\} \rightarrow \{\text{pickle}\}$ obtained from the transactions of a supermarket indicates that if a customer buys hotdogs and bread together, she is likely to buy pickles. To model the above likelihood, the confidence [12] of a rule $\{\text{hotdog}, \text{bread}\} \rightarrow \{\text{pickle}\}$ is the proportion of the transactions that have hotdogs and bread also include pickles. It has been regarded as the conditional probability that a customer buying *both* hotdogs *and* bread would trigger the additional purchase of pickles. To model the above rule in a graph, a possible way is to use two separate *edges* (see Figure 2; one from hotdog to pickle, and the other from bread to pickle, respectively), while the probability associated with each of these edges is the confidence of the rule. In the above graph model, however, *either one* of the hotdog or bread may trigger the purchase of pickle. This does not accurately express the intended condition of purchasing *both* the hotdog *and* bread. By contrast, the *hyperedges* in Graph Theory, by spanning multiple source nodes and one destination node, can model the above association rule (as illustrated in Figure 2). The probability associated with the hyperedge represents the likelihood of the purchase action denoted by the destination node when *all* purchase actions denoted by source nodes have happened.

On the other hand, in viral marketing, the traditional IC model activates a new node by the social influence probabilities associated with edges to the node. Aiming to capture both effects of item inference and social influence. We propose a new *Social Item Graph* (SIG). SIG models the likelihood for a purchase (or a set of purchases) to trigger another purchase in form of *hyperedges*, which may have one or multiple source nodes leading to one destination node. We define a social item graph as follows.

DEFINITION 4. Given a social graph of customers $G = (V, E)$ and a commodity item list I , a social item graph is denoted by $G_{SI} = (V_{SI}, E_H)$, where V_{SI} is a set of purchase actions and E_H is a set of hyperedges over V_{SI} . A node $n \in V_{SI}$ is denoted as (v, i) , where $v \in V$ and $i \in I$. A hyperedge $e \in E_H$ is of the following form:

$$\{(u_1, i_1), (u_2, i_2), \dots, (u_m, i_m)\} \rightarrow (v, i)$$

where u_i is in the neighborhood of v in G , i.e., $u_i \in N_G(v) = \{u | d(u, v) \leq 1\}$.⁶

Note that the conventional social influence edge in a social graph with one source and one destination can still be modeled in an SIG as a simple edge associated with a corresponding influence probability. Nevertheless, the influence probability from a person to another can vary for different items (e.g., a person’s influence on another person for cosmetics and smartphones may vary.). Moreover, although an SIG may model the purchases more accurately with the help of *both* social influence and item inference, the complexity of processing an SIG with hyperedges is much higher than simple edges in the traditional social graph that denotes only social influence.

For simplicity, let \mathbf{u} and \mathbf{v} (i.e., the symbols in Typewriter style) represent the nodes (u, i) and (v, i) in SIG for the rest of this paper. We also denote a hyperedge as $e \equiv \mathbf{U} \rightarrow \mathbf{v}$, where \mathbf{U} is a set of source nodes and \mathbf{v} is the destination node. Let the associated edge weight be p_e , which represents the *activation probability* for \mathbf{v} to be activated if all source nodes in \mathbf{U} are activated. Note that the activation probability is for one single hyperedge $\mathbf{U} \rightarrow \mathbf{v}$. Other hyperedges sharing the same destination may have different activation probabilities. For example, part of the source nodes in a hyperedge $\{\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}\} \rightarrow \mathbf{x}$ can still activate \mathbf{x} , e.g., by $\{\mathbf{a}, \mathbf{b}, \mathbf{c}\} \rightarrow \mathbf{x}$ with a different hyperedge with its own activation probability.

3.2 Diffusion Process in Social Item Graph

Next we introduce the diffusion process in SIG, which is inspired by the probability-based approach behind *Independent Cascade* (IC) to captures the word-of-mouth behavior in the real world [7].⁷ This diffusion process fits the item inferences captured in an SIG naturally, as we can derive conditional probabilities on hyperedges to describe the trigger (activation) of purchase actions on a potential purchase.

The diffusion process in SIG starts with all nodes inactive initially. Let \mathbf{S} denote a set of seeds (purchase actions). Let a node $\mathbf{s} \in \mathbf{S}$ be a seed. It immediately becomes active. Given all the nodes in a source set \mathbf{U} at iteration $\iota - 1$, if they are all active at iteration ι , a hyperedge $e \equiv \mathbf{U} \rightarrow \mathbf{v}$ has a chance to activate the inactive \mathbf{v} with probability p_e . Each node (v, i) can be activated once, but it can try to activate other nodes multiple times, one for each incident hyperedges. For the seed selection problem that we target on, the total number of activated nodes represents the number of items adopted by customers (called *total adoptions* for the rest of this paper).

4. SOCIAL ITEM MAXIMIZATION

⁶Notice that when $u_1 = u_2 = \dots = u_m = v$, the hyperedge represents the item inference of item i . On the other hand, when $i_1 = i_2 = \dots = i_m = i$, it becomes the social influence of user u on v .

⁷Several variants of IC model have been proposed [6, 5]. However, they focus on modeling the diffusion process between users, such as aspect awareness [6], which is not suitable for social item graph since the topic is embedded in each SIG node.

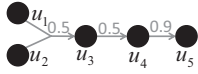


Figure 3: A non-submodular example

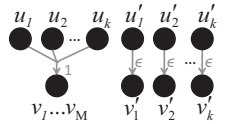
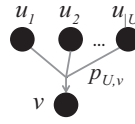
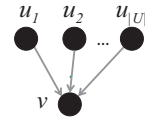


Figure 4: An example of SIMP



(a) Original



(b) Transformed

Figure 5: An illustration of graph transformations

Upon the proposed Social Item Graph (SIG), we now formulate a new seed selection problem, called *Social Item Maximization Problem* (SIMP), that selects a set of seed purchase actions to maximize potential sales or revenue in a marketing campaign. In Section 5, we will describe how to construct the SIG from purchase logs by a machine learning approach.

DEFINITION 5. *Given a seed number k , a list of targeted items I , and a social item graph $G_{SI}(V_{SI}, E_H)$, SIMP selects a set S of k seeds in V_{SI} such that $\alpha_{G_{SI}}(S)$, the total adoption function of S , is maximized.*

Note that a seed in SIG represents the adoption/purchase action of a specific item by a particular customer. The total adoption function $\alpha_{G_{SI}}$ represents the total number of product items ($\in I$) purchased. By assigning prices to products and costs to the selected seeds, an extension of SIMP is to maximize the total revenue subtracted by the cost.

Here we first discuss the challenges in solving SIMP before introducing our algorithm. Note that, for the influence maximization problem based on the IC model, Kempe et al. propose a $1 - 1/e$ approximation algorithm [17], thanks to the submodularity in the problem. Unfortunately, the submodularity does not hold for the total adoption function $\alpha_{G_{SI}}(S)$ in SIMP. Specifically, if the function $\alpha_{G_{SI}}(S)$ satisfies the submodularity, for any node i and any two subsets of nodes S_1 and S_2 where $S_1 \subseteq S_2$, $\alpha_{G_{SI}}(S_1 \cup \{i\}) - \alpha_{G_{SI}}(S_1) \geq \alpha_{G_{SI}}(S_2 \cup \{i\}) - \alpha_{G_{SI}}(S_2)$ should hold. However, a counter example is illustrated below.

EXAMPLE 1. *Consider an SIMP instance with a customer and five items in Figure 3. Consider the case where $S_1 = \{u_4\}$, $S_2 = \{u_1, u_4\}$, and i corresponds to node u_2 . For seed sets $\{u_4\}$, $\{u_2, u_4\}$, $\{u_1, u_4\}$ and $\{u_1, u_2, u_4\}$, $\alpha_{G_{SI}}(\{u_4\}) = 1.9$, $\alpha_{G_{SI}}(\{u_2, u_4\}) = 2.9$, $\alpha_{G_{SI}}(\{u_1, u_4\}) = 2.9$, and $\alpha_{G_{SI}}(\{u_1, u_2, u_4\}) = 4.4$. Thus, $\alpha_{G_{SI}}(S_1 \cup \{u_2\}) - \alpha_{G_{SI}}(S_1) = 1 < 1.5 = \alpha_{G_{SI}}(S_2 \cup \{u_2\}) - \alpha_{G_{SI}}(S_2)$. Hence, the submodularity does not hold.*

Since the submodularity does not exist in SIMP, the $1 - 1/e$ approximation ratio of the greedy algorithm in [17] does not hold. Now, an interesting question is how large the ratio becomes. Example 2 shows an SIMP instance where the greedy algorithm performs poorly.

EXAMPLE 2. *Consider an example in Figure 4, where nodes v_1, v_2, \dots, v_M all have a hyperedge with the probability as 1 from the same k sources u_1, u_2, \dots, u_k , and ϵ is an arbitrarily small edge probability $\epsilon > 0$. The greedy algorithm selects one node in each iteration, i.e., it selects u'_1, u'_2, \dots, u'_k as the seeds with a total adoption $k + k\epsilon$. However, the optimal solution actually selects u_1, u_2, \dots, u_k as the seeds and results in the total adoption $M + k$. Therefore, the approximation ratio of the greedy algorithm is at least $(M + k)/(k + k\epsilon)$, which is close to M/k for a large M , where M could approach $|V_{SI}|$ in the worst case.*

One may argue that the above challenges in SIMP may be alleviated by transforming G_{SI} into a graph with only simple edges, as displayed in Figure 5, where the weight of every $u_i \rightarrow v$ with $u_i \in U$ can be set independently. However, if a source node $u_m \in U$ of v is difficult to activate, the probability for v to be activated approaches zero in Figure 5 (a)

due to u_m . However, in Figure 5 (b), the destination v is inclined to be activated by sources in U , especially when U is sufficiently large. Thus, the idea of graph transformation does not work.

4.1 Hyperedge-Aware Greedy (HAG)

Here, we propose an algorithm for SIMP, *Hyperedge-Aware Greedy (HAG)*, with performance guarantee. The approximation ratio is proved in Section 4.3. A hyperedge requires all its sources activated first in order to activate the destination. Conventional single node greedy algorithms perform poorly because hyperedges are not considered. To address this important issue, we propose *Hyperedge-Aware Greedy (HAG)* to select multiple seeds in each iteration.

A naive algorithm for SIMP would examine, $C_k^{|V_{SI}|}$ combinations are involved to choose k seeds. In this paper, as multiple seeds tend to activate all source nodes of a hyperedge in order to activate its destination, an effective way is to consider only the combinations which include the source nodes of any hyperedge. We call the source nodes of a hyperedge as a *source combination*. Based on this idea, in each iteration, HAG includes the source combination leading to *the largest increment on total adoption divided by the number of new seeds added in this iteration*. Note that only the source combinations with no more than k sources are considered. The iteration continues until k seeds are selected. Note that HAG does not restrict the seeds to be the source nodes of hyperedges. Instead, the source node u of any simple edge $u \rightarrow v$ in SIG is also examined.

Complexity of HAG. To select k seeds, HAG takes at most k rounds. In each round, the source combinations of $|E_H|$ hyperedges are tried one by one, and the diffusion cost is c_{dif} , which will be analyzed in Section 4.2. Thus, the time complexity of HAG is $O(k \times |E_H| \times c_{dif})$.

4.2 Acceleration of Diffusion Computation

To estimate the total adoption for a seed set, it is necessary to perform Monte Carlo simulation based on the diffusion process described in Section 3.2 for many times. Finding the total adoption is very expensive, especially when a node v can be activated by a hyperedge with a large source set U , which indicates that there also exist many other hyperedges with an arbitrary subset of U as the source set to activate v . In other words, enormous hyperedges need to be examined for the diffusion on an SIG. It is essential to reduce the computational overhead. To address this issue, we propose a new index structure, called *SIG-index*, by exploiting FP-Tree [12] to pre-process source combinations in hyperedges in a compact form in order to facilitate efficient derivation of activation probabilities during the diffusion process.

The basic idea behind SIG-index is as follows. For each node v with the set of activated in-neighbors $N_{v,t}^a$ in iteration t , if v has not been activated before t , the diffusion process will try to activate v via every hyperedge $U \rightarrow v$ where the last source in U has been activated in iteration $t-1$. To derive the activation probability of a node v from the weights of hyperedges associated with v , we first define the activation probability as follows.

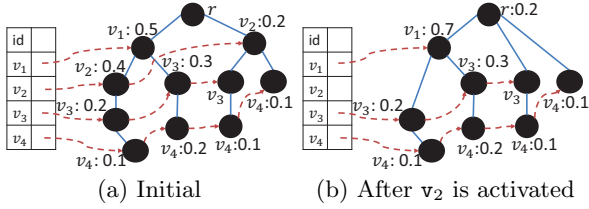


Figure 6: An illustration of SIG-index

DEFINITION 6. The activation probability of v at ι is

$$ap_{v,\iota} = 1 - \prod_{U \rightarrow v \in E_H, U \subseteq N_{v,\iota-1}, U \not\subseteq N_{v,\iota-2}} (1 - p_{U \rightarrow v}).$$

where $N_{v,\iota-1}$ and $N_{v,\iota-2}$ denote the set of active neighbors of v in iteration $\iota - 1$ and $\iota - 2$, respectively.

The operations on an SIG-index occur two phases: Index Creation Phase and Diffusion Processing Phase. As all hyperedges satisfying Definition 6 must be accessed, the SIG-index stores the hyperedge probabilities in Index Creation Phase. Later, the SIG-index is updated in Diffusion Processing Phase to derive the activation probability efficiently.

Index Creation Phase. For each hyperedge $U \rightarrow v$, we first regard each source combination $U = \{v_1, \dots, v_{|U|}\}$ as a transaction to build an FP-tree [12] by setting the minimum support as 1. As such, $v_1, \dots, v_{|U|}$ forms a path $r \rightarrow v_1 \rightarrow v_2 \dots \rightarrow v_{|U|}$ from the root r in the FP-tree to node $v_{|U|}$ in U . Different from the FP-Tree, the SIG-index associates the probability of each hyperedge $U \rightarrow v$ with the last source node $v_{|U|}$ in U .⁸ Initially the probability associated with the root r is 0. Later the the SIG-index is updated during the diffusion process. Example 3 illustrates the SIG-index created based on an SIG.

EXAMPLE 3. Consider an SIG graph with five nodes, v_1 - v_5 , and nine hyperedges with their associated probabilities in parentheses: $\{v_1\} \rightarrow v_5$ (0.5), $\{v_1, v_2\} \rightarrow v_5$ (0.4), $\{v_1, v_2, v_3\} \rightarrow v_5$ (0.2), $\{v_1, v_2, v_3, v_4\} \rightarrow v_5$ (0.1), $\{v_1, v_3\} \rightarrow v_5$ (0.3), $\{v_1, v_3, v_4\} \rightarrow v_5$ (0.2), $\{v_2\} \rightarrow v_5$ (0.2), $\{v_2, v_3, v_4\} \rightarrow v_5$ (0.1), $\{v_2, v_4\} \rightarrow v_5$ (0.1). Figure 6 (a) shows the SIG-index initially created for node v_5 .

Diffusion Processing Phase. The activation probability in an iteration is derived by traversing the initial SIG-index, which takes $O(|E_H|)$ time. However, a simulation may iterate a lot of times. To further accelerate the traversing process, we adjust the SIG-index for the activated nodes in each iteration. More specifically, after a node v^a is activated, accessing an hyperedge $U \rightarrow v$ with $v^a \in U$ becomes easier since the number remaining inactivated nodes in $U - \{v^a\}$ is reduced. Accordingly, SIG-index is modified by traversing every vertex labeled as v^a on the SIG-index in the following steps. 1) If v^a is associated with a probability p_a , it is crucial to aggregate the old activation probabilities p_a of v^a and p_p of its parent v^p , and update activation probability associated with v^p as $1 - (1 - p_a)(1 - p_p)$, since the source combination needed for accessing the hyperedges associated with v^a and v^p becomes the same. The aggregation is also performed when v^p is r . 2) If v^a has any children c , the parent of c is changed to be v^a , which removes the processed v^a from the index. 3) After processing every node v^a in the SIG-index, we obtain the activation probability of v in the root r . After the probability is accessed for activating v , the probability of r is reset to 0 for next iteration.

⁸For ease of explanation, we assume the order of nodes in the SIG-index follows the ascending order of subscript.

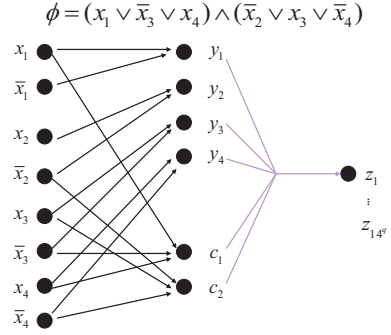


Figure 7: An illustration instance built for 3-SAT

EXAMPLE 4. Consider an example with v_2 activated in an iteration. To update the SIG-index, each vertex v_2 in Figure 6 (a) is examined by traversing the linked list of v_2 . First, the left vertex with label v_2 is examined. SIG-index reassigns the parent of v_2 's child (labeled as v_3) to the vertex labeled as v_1 , and aggregate the probability 0.4 on the v_2 and 0.5 on vertex v_1 , since the hyperedge $\{v_1, v_2\} \rightarrow v_5$ can be accessed if the node v_1 is activated later. The probability of v_1 becomes $1 - (1 - p_{v_1})(1 - p_{v_2}) = 0.7$. Then the right vertex with label v_2 is examined. The parent of its two children is reassigned to the root r . Also, the probability of itself (0.2) is aggregated with the root r , indicating that the activation probability of node v_5 in the next iteration is 0.2.

Complexity Analysis. For Index Creation Phase, the initial SIG-index for v is built by examining the hyperedges two times with $O(|E_H|)$ time. The number of vertices in SIG-index is at most $O(c_d |E_H|)$, where c_d is the number of source nodes in the largest hyperedge. During Diffusion Processing Phase, each vertex in SIG-index is examined only once through the node-links, and the parent of a vertex is changed at most $O(c_d)$ times. Thus, the overall time to complete a diffusion requires at most $O(c_d |E_H|)$ time.

4.3 Hardness Results

From the discussion earlier, it becomes obvious that SIMP is difficult. In the following, we will prove that SIMP is inapproximable with a non-constant ratio n^c for all $c < 1$, with a gap-introducing reduction from an NP-complete problem 3-SAT to SIMP, where n is the number of nodes in an SIG. Note that the theoretical result only shows that for any algorithm, there exists a problem instance of SIMP (i.e., a pair of an SIG graph and a seed number k) that the algorithm can not obtain a solution better than $1/n$ times the optimal solution. It does not imply that an algorithm always performs badly in every SIMP instance.

LEMMA 1. For a positive integer q , there is a gap-introducing reduction from 3-SAT to SIMP, which transforms an n_{var} -variables expression ϕ to an SIMP instance with the SIG as $G_{SI}(V_{SI}, E_H)$ and the k as n_{var} such that

- if ϕ is satisfiable, $\alpha_{G_{SI}}^* \geq (m_{cla} + 3n_{var})^q$, and
- if ϕ is not satisfiable, $\alpha_{G_{SI}}^* < m_{cla} + 3n_{var}$,

where $\alpha_{G_{SI}}^*$ is the optimal solution of this instance, n_{var} is the number of Boolean variables, and m_{cla} is the number of clauses. Hence there is no $(m_{cla} + 3n_{var})^{q-1}$ approximation algorithm for SIMP unless $P = NP$.

PROOF. Given a positive integer q , for an instance ϕ of 3-SAT with n_{var} Boolean variables $a_1, \dots, a_{n_{var}}$ and m_{cla} clauses $C_1, \dots, C_{m_{cla}}$, we construct an SIG G_{SI} with three node sets X, Y and Z as follows. 1) Each Boolean variable a_i corresponds to two nodes x_i, \bar{x}_i in X and one node y_i in Y . 2) Each clause C_k corresponds to one node c_k in Y . 3)

Z has $(|X| + |Y|)^q$ nodes. (Thus, G_{SI} has $(m_{cla} + 3n_{var})^q + m_{cla} + 3n_{var}$ nodes.) 4) For each y_j in Y, we add direct edges $x_j \rightarrow y_j$ and $\bar{x}_j \rightarrow y_j$. 5) For each c_k in Y, we add direct edges $\alpha \rightarrow c_k$, $\beta \rightarrow c_k$ and $\gamma \rightarrow c_k$, where α, β, γ are the nodes in X corresponding to the three literals in C_k . 6) We add a hyperedge $Y \rightarrow z_v$ from all for every $z_v \in Z$. The probability of every edge is set to 1. An example is illustrated in Figure 7.

We first prove that ϕ is satisfiable if and only if G_{SI} has a seed set S with n_{var} seeds and the total adoption of S contains Y. If ϕ is satisfiable, there exists a truth assignment T on Boolean variables $a_1, \dots, a_{n_{var}}$ satisfying all clauses of ϕ . Let $S = \{x_i | T(a_i) = 1\} \cup \{\bar{x}_j | T(a_j) = 0\}$, and S then has n_{var} nodes and the total adoption of S contains Y. On the other hand, if ϕ is not satisfiable, apparently there exists no seed set S with exactly one of x_i or \bar{x}_i selected for every i such that the total adoption of S contains Y. For other cases, 1) all seeds are placed in X, but there exists at least one i with both x_i and \bar{x}_i selected. In this case, there must exist some j such that none of x_j or \bar{x}_j are selected (since the seed number is n_{var}), and thus Y is not covered by the total adoption of S. 2) A seed is placed in Y. In this case, the seed can be moved to an adjacent x_i without reducing the total adoption. Nevertheless, as explained above, there exists no seed set S with all seeds placed in X such that the total adoption of S contains Y, and thus the total adoption of any seed set with a seed placed in Y cannot cover Y, either. With above observations, if ϕ is not satisfiable, G_{SI} does not have a seed set S with n_{var} seeds such that the total adoption of S contains Y. Since the nodes of Z can be activated if and only if the total adoption of S contains Y if and only if ϕ is satisfiable, we have

- if ϕ is satisfiable, $\alpha_{G_{SI}}^* \geq (m_{cla} + 3n_{var})^q$, and
- if ϕ is not satisfiable, $\alpha_{G_{SI}}^* < m_{cla} + 3n_{var}$.

The lemma follows. \square

THEOREM 1. *For any $\epsilon > 0$, there is no $n^{1-\epsilon}$ approximation algorithm for SIMP, assuming $P \neq NP$.*

PROOF. For any arbitrary $\epsilon > 0$, we set $q \geq \frac{2}{\epsilon}$. Then, by Lemma 1, there is no $(m_{cla} + 3n_{var})^{q-1}$ approximation algorithm for SIMP unless $P = NP$. Then $(m_{cla} + 3n_{var})^{q-1} \geq 2(m_{cla} + 3n_{var})^{q-2} \geq 2(m_{cla} + 3n_{var})^{q(1-\epsilon)} \geq (2(m_{cla} + 3n_{var})^q)^{1-\epsilon} \geq n^{1-\epsilon}$. Since ϵ is arbitrarily small, thus for any $\epsilon > 0$, there is no $n^{1-\epsilon}$ approximation algorithm for SIMP, assuming $P \neq NP$. The theorem follows. \square

With Theorem 3, no algorithm can achieve an approximation ratio better than n . In Theorem 1, we prove that SIG-index is correct, and HAG with SIG-index achieves the best ratio, i.e., it is n -approximated to SIMP. Note that the approximation ratio only guarantees the lower bound of total adoption obtained by HAG theoretically. Later in Section 6.3, we empirically show that the total adoption obtained by HAG is comparable to the optimal solution.

THEOREM 2. *HAG with SIG-index is n -approximated, where n is the number of nodes in SIG.*

PROOF. First, we prove that SIG-index obtains $ap_{v,\iota}$ correctly. Assume that there exists an incorrect $ap_{v,\iota}$, i.e., there exists an hyperedge $U \rightarrow v$ satisfying the conditions in Definition 6 (i.e., $U \not\subseteq N_{v,\iota-2}$ and $U \subseteq N_{v,\iota-1}$) but its probability is not aggregated to r in ι . However, the probability can not be aggregated before ι since $U \not\subseteq N_{v,\iota-2}$ and it must be aggregated no later than ι since $U \subseteq N_{v,\iota-1}$. There is a contradiction.

Proving that HAG with SIG-index is an n -approximation algorithm is simple. The upper bound of total adoption for the optimal algorithm is n , while the lower bound of the total

adoption for HAG is 1 because at least one seed is selected. In other words, designing an approximation algorithm for SIMP is simple, but it is much more difficult to have the hardness result for SIMP, and we have proven that SIMP is inapproximable within $n^{1-\epsilon}$ for any arbitrarily small ϵ . \square

THEOREM 3. *For any $\epsilon > 0$, there is no $n^{1-\epsilon}$ approximation algorithm for SIMP, assuming $P \neq NP$.*

PROOF. For any arbitrary $\epsilon > 0$, we set $q \geq \frac{2}{\epsilon}$. Then, by Lemma 1, there is no $(m_{cla} + 3n_{var})^{q-1}$ approximation algorithm for SIMP unless $P = NP$. Then $(m_{cla} + 3n_{var})^{q-1} \geq 2(m_{cla} + 3n_{var})^{q-2} \geq 2(m_{cla} + 3n_{var})^{q(1-\epsilon)} \geq (2(m_{cla} + 3n_{var})^q)^{1-\epsilon} \geq n^{1-\epsilon}$. Since ϵ is arbitrarily small, thus for any $\epsilon > 0$, there is no $n^{1-\epsilon}$ approximation algorithm for SIMP, assuming $P \neq NP$. The theorem follows. \square

COROLLARY 1. *HAG with SIG-index is n -approximated, where n is the number of nodes in SIG, because SIG-index only improves the efficiency.*

5. CONSTRUCTION OF SIG

To select seeds for SIMP, we need to construct the SIG from purchase logs and the social network. We first create possible hyperedges by scanning the purchase logs. Let τ be the timestamp of a given purchase $v = (v, i)$. v 's friends purchase and her own purchases that have happened within a given period before τ are considered as candidate source nodes to generate hyperedges to v . For each hyperedge e , the main task is then the estimation of its their activation probability p_e . Since p_e is unknown, it is estimated by maximizing the likelihood function based on observations in the purchase logs. Note that learning the activation probability p_e for each hyperedge e faces three challenges.

C1. Unknown distribution of p_e . How to properly model p_e is critical.

C2. Unobserved activations. When v is activated at time τ , this event only implies that at least one hyperedge successfully activates v before τ . It remains unknown which or hyperedge(s) actually triggers v , i.e., it may be caused by either the item inference or social influence or both. Therefore, we cannot simply employ the confidence of an association-rule as the corresponding hyperedge probability.

C3. Data Sparsity. The number of activations for a user to buy an item is small, whereas the number of possible hyperedge combinations is large. Moreover, new items emerge every day in e-commerce websites, which incurs the notorious cold-start problem. Hence, a method to deal with the data sparsity issue is necessary to properly model a SIG.

To address these challenges, we exploit a statistical inference approach to identify those hyperedges and learn their weights. In the following, we first propose a model of the edge function (to address the first challenge) and then exploit the smoothed expectation and maximization (EMS) algorithm [21] to address the second and third challenges.

5.1 Modeling of Hyperedge Probability

To overcome the first challenge, one possible way is to model the number of success activations and the number of unsuccessful activations by the binomial distributions. As such, p_e is approximated by the ratio of the number of success activations and the number of total activation trials. However, the binomial distribution function is too complex for computing the maximum likelihood of a vast number of data. To handle big data, previous study reported [14] that the binomial distribution (n, p) can be approximated by the Poisson distribution $\lambda = np$ when the time duration is sufficiently large. According to the above study, it is assumed

that the number of activations of a hyperedge e follows the Poisson distribution to handle the social influence and item inference jointly. The expected number of events equals to the intensity parameter λ . Moreover, we use an inhomogeneous Poisson process defined on the space of hyperedges to ensure that p_e varies with different e .

In the following, a hyperedge is of size n , if the cardinality of its source set \mathbf{U} is n . We denote the intensity of the number of activation trials of the hyperedge e as $\lambda_T(e)$. Then the successful activations of hyperedge e follows another Poisson process where the intensity is denoted by $\lambda_A(e)$. Therefore, the hyperedge probability p_e can be derived by parameters $\lambda_A(e)$ and $\lambda_T(e)$, i.e., $p_e = \frac{\lambda_A(e)}{\lambda_T(e)}$.

The maximum likelihood estimation can be employed to derive $\lambda_T(e)$. Nevertheless, $\lambda_A(e)$ cannot be derived as explained in the second challenge. Therefore, we use the expectation maximization (EM) algorithm, which is an extension of maximum likelihood estimation containing latent variables to $\lambda_A(e)$ which is modeled as the latent variable. Based on the observed purchase logs, the E-step first derives the likelihood Q-function of parameter p_e with $\lambda_A(e)$ as the latent variables. In this step, the purchase logs and p_e are given to find the probability function describing that all events on e in the logs occur according to p_e , whereas the probability function (i.e., Q-function) explores different possible values on latent variable $\lambda_A(e)$. Afterward, the M-step maximizes the Q-function and derives the new p_e for E-Step in the next iteration. These two steps are iterated until convergence.

With the employed Poisson distribution and EM algorithm, data sparsity remains an issue. Therefore, we further exploit a variant of EM algorithm, called *EMS algorithm* [21], to alleviate the sparsity problem by estimating the intensity of Poisson process using *similar hyperedges*. The parameter smoothing after each iteration is called S-Step, which is incorporated in EMS algorithm, in addition to the existing E-Step and M-Step.

5.2 Model Learning by EMS Algorithm

Let p_e and \hat{p}_e denote the true probability and estimated probabilities for hyperedge e in the EMS algorithm, respectively, where $e = U \rightarrow v$. Let N_U and K_e denote the number of activations of source set U in the purchase logs and the number of successful activations on hyperedge e , respectively. The EM algorithm is exploited to find the maximum likelihood of p_e , while $\lambda_A(e)$ is the latent variable because K_e cannot be observed (i.e., only N_U can be observed). Therefore, E-Step derives the likelihood function for $\{p_e\}$ (i.e., the Q-function) as follows,

$$Q(p_e, \hat{p}_e^{(i-1)}) = E_{K_e}[\log P(K_e, N_U | p_e) | N_U, \hat{p}_e^{(i-1)}], \quad (1)$$

where $\hat{p}_e^{(i-1)}$ is the hyperedge probability derived in the previous iteration, Note that N_U and $p_e^{(i-1)}$ are given parameters in iteration i , whereas p_e is a variable in the Q-function, and K_e is a random variable governed by the distribution $P(K_e | N_U, p_e^{(i-1)})$. Since p_e is correlated to $\lambda_T(U)$ and $\lambda_A(e)$, we derive the likelihood $P(K_e, N_U | p_e)$ as follows.

$$\begin{aligned} & P(K_e, N_U | p_e) \\ &= P(\{K_e\}_{e \in E_H}, \{N_U\}_{U \subseteq V_{SI}} | \{p_e\}_{e \in E_H}, \{\lambda_T(U)\}_{U \subseteq V_{SI}}) \\ &= P(\{K_e\}_{e \in E_H} | \{p_e\}_{e \in E_H}, \{N_U, \lambda_T(U)\}_{U \subseteq V_{SI}}) \\ & \quad \times P(\{N_U\}_{U \subseteq V_{SI}} | \{\lambda_T(U)\}_{U \subseteq V_{SI}}). \end{aligned}$$

It is assumed that $\{K_e\}$ is independent with $\{N_U\}$, and $Q(p_e, \hat{p}_e^{(i-1)})$ can be derived as follows:

$$\sum_{e \in E_H} \log P(K_e | N_U, p_e) + \log P(\{N_U\}_{U \subseteq V_{SI}} | \{\lambda_T(e)\}_{U \subseteq V_{SI}}).$$

Since only the first term contains the hidden K_e , only this term varies in different iterations of the EMS algorithm, because $\{N_U\}_{U \subseteq V_{SI}}$ in the second term always can be derived by finding the maximum likelihood as follows. Let $p_{U,k}$ denote the probability that the source set U exactly tries to activate the destination node k times, i.e., $p_{U,k} = P\{N_U = k\}$. The log-likelihood of λ_T is

$$\begin{aligned} & \sum_k p_{U,k} \ln\left(\frac{\lambda_T^k e^{-\lambda_T}}{k!}\right) = \sum_k p_{U,k} (-\lambda_T + k \ln \lambda_T - \ln k!) \\ &= -\lambda_T + (\ln \lambda_T) \sum_k k p_{U,k} - \sum_k p_{U,k} \ln k!. \end{aligned}$$

We acquire the maximum likelihood by finding the derivative with regard to λ_T :

$$-1 + \frac{1}{\lambda_T} \sum_k k p_{U,k} = 0. \quad (2)$$

Thus, the maximum log-likelihood estimation of $\lambda_T = \sum_k k p_{U,k}$, representing that the expected activation times (i.e., $\hat{\lambda}_T(e)$) is N_U . Let $\mathcal{A} = \{(v, \tau)\}$ denote the action log set, where each log (v, τ) represents that v is activated at time τ . N_U is calculated by scanning \mathcal{A} and find the times that all the nodes in \mathbf{U} are activated.

Afterward, we focus on the first term of $Q(p_e, \hat{p}_e^{(i-1)})$. Let $p_{e,k} = P\{K_e = k\}$ denote the probability that the hyperedge e exactly activates the destination node k times. In E-step, we first find the expectation for K_e as follows.

$$\begin{aligned} & \sum_{e \in E_H} \sum_{k=1, \dots, N_U} p_{e,k} \log(k | N_U, p_e) \\ &= \sum_{k=1, \dots, N_U} p_{e,k} \log P(k | N_U, p_e) \\ &= \sum_{k=1, \dots, N_U} p_{e,k} \log\left(\binom{N_U}{k} p_e^k (1-p_e)^{N_U-k}\right) \\ &= \sum_{k=1, \dots, N_U} p_{e,k} \left(\log\left(\binom{N_U}{k}\right) + k \log p_e + (N_U - k) \log(1-p_e) \right). \end{aligned}$$

Since $\sum_{k=1, \dots, N_U} p_{e,k} k = E[K_e]$ and $\sum_{k=1, \dots, N_U} p_{e,k} = 1$, the log-likelihood of the first term is further simplified as

$$\sum_{k=1, \dots, N_U} p_{e,k} \log\left(\binom{N_U}{k}\right) + N_U \log(1-p_e) + E[K_e](\log p_e - \log(1-p_e)).$$

Afterward, M-step maximizes the Q-function by finding the derivative with regard to p_e :

$$\frac{-N_U}{1-p_e} + E[K_e] \left(\frac{1}{p_e} + \frac{1}{1-p_e} \right) = 0$$

$$p_e = E[K_e] / N_U$$

Therefore, the maximum likelihood estimator \hat{p}_e is $\frac{E[K_e]}{N_U}$, $\hat{\lambda}_T(U)$ is N_U , and $\hat{\lambda}_A(e) = E[K_e]$.

The problem remaining is to take expectation of the latent variables $\{K_e\}$ in E-step. Let $\{w_{e,a}\}_{e \in E_H, a = (v, \tau) \in \mathcal{A}}$ be the conditional probability that v is activated by the source set U of e at τ given v is activated at τ , and let E_a denote the set of candidate hyperedges containing every possible e with its source set activated at time $\tau - 1$, i.e., $E_a = \{(\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_n) \rightarrow v | v_i = 1, \dots, n, \mathbf{u}_i \in N(\mathbf{v}_i), (\mathbf{u}_i, \tau - 1) \in \mathcal{A}\}$. It's easy to show that given the estimation of the probability of hyperedges, $w_{e,a} = \frac{\hat{p}_e}{1 - \prod_{e' \in E_a} (1 - \hat{p}_{e'})}$, since $1 - \prod_{e' \in E_a} (1 - \hat{p}_{e'})$ is the probability for \mathbf{v} to be activated by any hyperedge at time τ . The expectation of

K_e is $\sum_{a \in A, e \in E_a \cap E_{H,n}} w_{e,a}$, i.e., the sum of expectation of each successful activation of \mathbf{v} from hyperedge e , and $E_{H,n} = \{(\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_n; \mathbf{v}) \in E_H\}$ contains all size n hyperedges.

To address the data sparsity problem, we leverage information from similar hyperedges (described later). Therefore, our framework includes an additional step to smooth the results of M-Step. Kernel smoothing is employed in S-Step. In summary, we have the following steps:

E-Step:

$$E[K_e] = \sum_{a \in A, e \in E_a \cap E_{H,n}} w_{e,a},$$

$$w_{e,a} = \frac{\hat{p}_e}{1 - \prod_{e' \in E_a} (1 - \hat{p}_{e'})}.$$

M-Step:

$$p_e = \frac{\sum_{a \in A, e \in E_a \cap E_{H,n}} w_{e,a}}{N_U},$$

$$\lambda_A(e) = \sum_{a \in A, e \in E_a \cap E_{H,n}} w_{e,a},$$

$$\lambda_T(U) = N_U.$$

S-Step: To address the data sparsity problem, we leverage information from similar hyperedges (described later). Therefore, in addition to E-Step and M-Step, EMS includes **S-Step**, which smooths the results of M-Step. Kernel smoothing is employed in S-Step as follows:

$$\hat{\lambda}_A(e) = \sum_{a \in A, e' \in E_a \cap E_{H,n}} w_{e',a} L_h(F(e) - F(e'))$$

$$\hat{\lambda}_T(U) = \sum_{U' \subseteq V_{SI}} N_U L_h(F(U) - F(U'))$$

where L_h is a kernel function with bandwidth h , and F is the mapping function of hyperedges, i.e., $F(e)$ maps a hyperedge e to a vector. The details of dimension reduction for calculating F to efficiently map hyperedges into Euclidean space are shown in [2] due to space constraint. If the hyperedges e and e' are similar, the distance of the vectors $F(e)$ and $F(e')$ is small. Moreover, a kernel function $L_h(x)$ is a positive function symmetric at zero which decreases when $|x|$ increases, and the bandwidth h controls the extent of auxiliary information taken from similar hyperedges.⁹ Intuitively, kernel smoothing can identify the correlation of \hat{p}_{e_1} with $e_1 = \mathbf{U}_1 \rightarrow \mathbf{v}_1$ and \hat{p}_{e_2} with $e_2 = \mathbf{U}_2 \rightarrow \mathbf{v}_2$ for nearby \mathbf{v}_1 and \mathbf{v}_2 and similar \mathbf{U}_1 and \mathbf{U}_2 .

6. EVALUATION

We conduct comprehensive experiments to evaluate the proposed SIG model, learning framework and seed selection algorithms. In Section 6.1, we discuss the data preparation for our evaluation. In Section 6.2, we compare the predictive power of the SIG model against two baseline models: i) independent cascade model learned by implementing Variance Regularized EM Algorithm (VAREM) [18] and ii) the generalized threshold (GT) model learned by [11].¹⁰ In addition, we evaluate the learning framework based on the proposed EM and EMS algorithms. Next, in Section 6.3, we evaluate the proposed HAG algorithm for SIMP in comparison to a number of baseline strategies, including random, single node selection, social, and item approaches. Finally, in

⁹A symmetric Gaussian kernel function is often used [13].

¹⁰<http://people.cs.ubc.ca/~welu/downloads.html>

Section 6.4, we evaluate alternative approaches for diffusion processing, which is essential and critical for HAG, based on SIG-index, Monte Carlo simulations and sorting enhancement.

6.1 Data Preparation

Here, we conduct comprehensive experiments using three real datasets to evaluate the proposed ideas and algorithms. The first dataset comes from Douban [1], a social networking website allowing users to share music and books with friends. Dataset *Douban* contains 5,520,243 users and 86,343,003 friendship links, together with 7,545,432 (user, music) and 14,050,265 (user, bookmark) pairs, representing the music noted and the bookmarks noted by each user, respectively. We treat those (user, music) and (user, bookmark) pairs as purchase actions. In addition to *Douban*, we adopt two public datasets, i.e., *Gowalla* and *Epinions*. Dataset *Gowalla* contains 196,591 users, 950,327 links, and 6,442,890 check-ins [8]. Dataset *Epinions* contains 22,166 users, 335,813 links, 27 categories of items, and 922,267 ratings with timestamp [22]. Notice that we do not have data directly reflecting item inferences in online stores, so we use the purchase logs for learning and evaluations. The experiments are implemented in an HP DL580 server with 4 Intel Xeon E7-4870 2.4 GHz CPUs and 1 TB RAM.

We split all three datasets into 5-fold, choose one subsample as training data, and test the models on the remaining subsamples. Specifically, we ignore the cases when the user and her friends did not buy anything. Finally, to evaluate the effectiveness of the proposed SIG model (and the learning approaches), we obtain the purchase actions in the following cases as the ground truth: 1) item inference - a user buys some items within a short period of time; and 2) social influence - a user buys an item after at least one of her friends bought the item. The considered periods of item inference and social influence are set differently according to [15] and [27], respectively. It is worth noting that only the hyperedges with the probability larger than a threshold parameter θ are considered. We empirically tune θ to obtain the default setting based on optimal F1-Score. Similarly, the threshold parameter θ for the GT model is obtained empirically. The reported precision, recall, and F1 are the average of these tests. Since both SIGs and the independent cascade model require successive data, we split the datasets into continuous subsamples.

6.2 Model Evaluation

Tables 1 present the precision, recall, and F1 of SIG, VAREM and GT on *Douban*, *Gowalla*, and *Epinions*. All three models predict most accurately on *Douban* due to the large sample size. The SIG model significantly outperforms the other two models on all three datasets, because it takes into account both effects of social influence and item inference, while the baseline models only consider the social influence. The difference of F1 score between SIG and baselines is more significant on *Douban*, because it contains more items. Thus, item influence plays a more important role. Also, when the user size increases, SIG is able to extract more social influence information leading to better performance than the baselines. The offline training time is 1.68, 1.28, and 4.05 hours on *Epinions*, *Gowalla*, *Douban*, respectively.

To evaluate the approaches adopted to learn the activation probabilities of hyperedges for construction of SIG, Fig. 8 compares the prevision and F1 of EMS and EM algorithms on *Epinions* (results on other datasets are consistent and thus not shown due to space limitation). Note that EM is a

Table 1: Comparison of precision, recall, and F1 for three models on Douban, Gowalla, Epinions

Dataset	Douban			Gowalla			Epinions		
	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score
GT	0.420916	0.683275	0.520927	0.124253	0.435963	0.171214	0.142565	0.403301	0.189999
VAREM	0.448542	0.838615	0.584473	0.217694	0.579401	0.323537	0.172924	0.799560	0.247951
SIG	0.869348	0.614971	0.761101	0.553444	0.746408	0.646652	0.510118	0.775194	0.594529

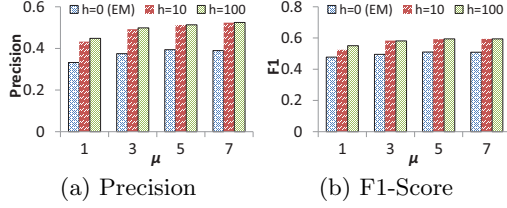


Figure 8: Comparisons of precision and F1 in various μ and h on *Epinions*

special case of EMS (with the smoothing parameter $h = 0$, i.e., no similar hyperedge used for smoothing). EMS outperforms EM on both precision and F1-score in all settings of μ (the maximum size of hyperedges) and h tested. Moreover, the precision and F1-score both increases with h as a larger h overcomes data sparsity significantly. As μ increases, more combinations of social influence and item inference can be captured. Therefore, the experiments show that a higher μ improves F1-score without degrades the precision. It manifests that the learned hyperedges are effective for predicting triggered purchases.

6.3 Algorithm Effectiveness and Efficiency

We evaluate HAG proposed for SIMP, by selecting top 10 items as the marketing items to measure their total adoption, in comparison with a number of baselines: 1) *Random approach (RAN)*. It randomly selects k nodes as seeds. Note that the reported values are the average of 50 random seed sets. 2) *Single node selection approach (SNS)*. It selects a node with the largest increment of the total adoption in each iteration, until k seeds are selected, which is widely employed in conventional seed selection problem [6, 7, 17]. 3) *Social approach (SOC)*. It only considers the social influence in selecting the k seeds. The hyperedges with nodes from different products are eliminated in the seed selection process, but they are restored for calculation of the final total adoption. 4) *Item approach (IOC)*. The seed set is the same as HAG, but the prediction is based on item inference only. For each seed set selected by the above approaches, the diffusion process is simulated 300 times. We report the average in-degree of nodes learned from the three datasets in the following: Douban is 39.56; Gowalla is 9.90; Epinions is 14.04. In this section, we evaluate HAG by varying the number of seeds (i.e., k) using two metrics: 1) total adoption, and 2) running time.

To understand the effectiveness, we first compared all those approaches with the optimal solution (denoted as OPT) in a small subgraph sampled, *Sample*, from the SIG of *Douban* with 50 nodes and 58 hyperedges. Figures 9 (a) displays the total adoption obtained by different approaches. As shown, HAG performs much better than the baselines and achieves comparable total adoption with OPT (the difference decreases with increased k). Note that OPT is not scalable as shown in Figures 9 (b) since it needs to examine all combination with k nodes. Also, OPT takes more than 1 day for selecting 6 seeds in *Sample*. Thus, for the rest of experiments, we exclude OPT.

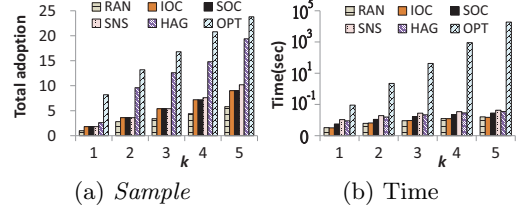


Figure 9: Total adopting and running time of *Sample* in various k

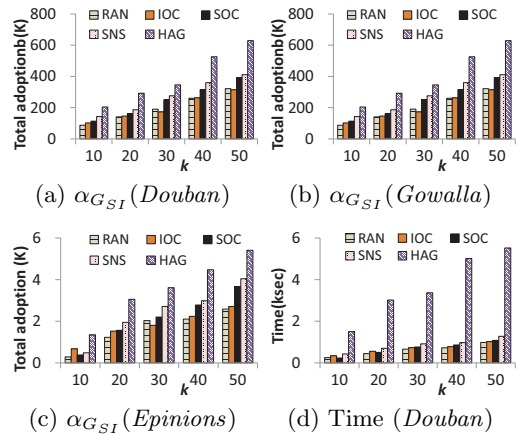


Figure 10: Total adoption α_{GSI} and running time in various k

Figures 10 (a)-(c) compare the total adoptions of different approaches in the SIG learnt from real networks. They all grow as k increases, since a larger k increases the chance for seeds to influence others to adopt items. Figure 10 (a)-(c) manifest that HAG outperforms all the other baselines for any k in SIG model. Among them, SOC fails to find good solutions since item inference is not examined during seed selection. IOC performs poorly without considering social influence. SNS only includes one seed at a time without considering the combination of nodes that may activate many other nodes via hyperedges.

Figure 10 (d) reports the running time of those approaches. Note that the trends upon *Gowalla* and *Epinions* are similar with *Douban*. Thus we only report the running time of *Douban* due to the space constraint. Taking the source combinations into account, HAG examines source combinations of hyperedges in E_H and obtains a better solution by spending more time since the number of hyperedges is often much higher than the number of nodes.

6.4 Online Diffusion Processing

Diffusion processing is an essential operation in HAG. We evaluate the efficiency of diffusion processing based on SIG-index (denoted as SX), in terms of the running time, in comparison with that based on the original Monte Carlo simulation (denoted as MC) and the sorting enhancement (denoted as SORTING), which accesses the hyperedges in descending order of their weights. Figure 11 plots the run-

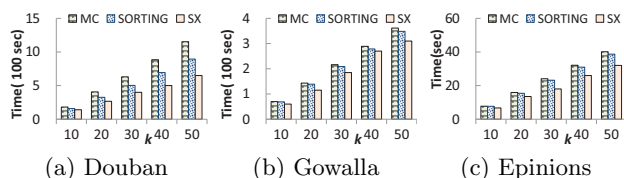


Figure 11: Running time of different simulation methods

ning time of SX, SORTING, and MC under various k using the *Douban*, *Gowalla*, and *Epinions*. For each k , the average running times of 50 randomly selected seed sets for SX, SORTING, and MC, are reported. The diffusion process is simulated 300 times for each seed set. As Figure 11 depicts, the running time for all the three approaches grows as k increases, because a larger number of seeds tends to increase the chance for other nodes to be activated. Thus, it needs more time to diffuse. Notice that SX takes much less time than SORTING and MC, because SX avoids accessing hyperedges with no source nodes newly activated while calculating the activation probability. Moreover, the SIG-index is updated dynamically according to the activated nodes in diffusion process. Also note that the improvement by MC over SORTING in *Douban* is more significant than that in *Gowalla* and *Epinions*, because the average in-degree of nodes is much larger in *Douban*. Thus, activating a destination at an early stage can effectively avoid processing many hyperedges later.

7. CONCLUSION

In this paper, we argue that existing techniques for item inference recommendation and seed selection need to jointly take social influence and item inference into consideration. We propose Social Item Graph (SIG) for capturing purchase actions and predicting potential purchase actions. We propose an effective machine learning approach to construct a SIG from purchase action logs and learn hyperedge weights. We also develop efficient algorithms to solve the new and challenging Social Item Maximization Problem (SIMP) that effectively select seeds for marketing. Experimental results demonstrate the superiority of the SIG model over existing models and the effectiveness and efficiency of the proposed algorithms for processing SIMP. We also plan to further accelerate the diffusion process by indexing additional information on SIG-index.

8. REFERENCES

- [1] The Douban dataset download. <http://arbor.ee.ntu.edu.tw/~hshshuai/Douban.tar.bz2>.
- [2] When social influence meets association rule learning. In *CoRR (1502.07439)*, 2016.
- [3] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *SIGMOD*, 1993.
- [4] R. M. Bond, C. J. Fariss, J. J. Jones, A. D. Kramer, C. Marlow, J. E. Settle, and J. H. Fowler. A 61-million-person experiment in social influence and political mobilization. *Nature*, 2012.
- [5] S. Bourigault, S. Lamprier, and P. Gallinari. Representation learning for information diffusion through social networks: an embedded cascade model. In *WSDM*, 2016.
- [6] S. Chen, J. Fan, G. Li, J. Feng, K. Tan, and J. Tang. Online topic-aware influence maximization. In *VLDB*, 2015.
- [7] W. Chen, C. Wang, and Y. Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *KDD*, 2010.
- [8] E. Cho, S. A. Myers, and J. Leskovec. Friendship and mobility: user movement in location-based social networks. In *KDD*, 2011.
- [9] P. Domingos and M. Richardson. Mining the network value of customers. In *KDD*, 2001.
- [10] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan. Learning influence probabilities in social networks. In *WSDM*, 2010.
- [11] A. Goyal, W. Lu, and L. V. S. Lakshmanan. SIMPATH: An efficient algorithm for influence maximization under the linear threshold model. In *ICDM*, 2011.
- [12] J. Han, M. Kamber, and J. Pei. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 3rd Edition, 2011.
- [13] N. L. Hjort and M. C. Jones. Locally parametric nonparametric density estimation. In *The Annals of Statistics*, 1996.
- [14] R. V. Hogg and E. A. Tanis. *Probability and statistical inference*. Prentice Hall, 7th Edition, 2005.
- [15] R. Jones and K. Klinkner. Beyond the session timeout: automatic hierarchical segmentation of search topics in query logs. In *CIKM*, 2008.
- [16] R. J. B. Jr. Efficiently mining long patterns from databases. In *SIGMOD*, 1998.
- [17] D. Kempe, J. M. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *KDD*, 2003.
- [18] H. Li, T. Cao, and Z. Li. Learning the information diffusion probabilities by using variance regularized em algorithm. In *ASONAM*, 2014.
- [19] J. Nail. The consumer advertising backlash, 2004. Forrester Research and Intelliseek Market Research Report.
- [20] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. In *ICDT*, 1999.
- [21] B. W. Silverman, M. C. Jones, J. D. Wilson, and D. W. Nychka. A smoothed em approach to indirect estimation problems, with particular reference to stereology and emission tomography. *Journal of the Royal Statistical Society. Series B (Methodological)*, 1990.
- [22] J. Tang, H. Gao, H. Liu, and A. D. Sarma. etrust: Understanding trust evolution in an online world. In *KDD*, 2012.
- [23] J. Tang, S. Wu, and J. Sun. Confluence: conformity influence in large social networks. In *KDD*, 2013.
- [24] Y. Tang, X. Xiao, and Y. Shi. Influence maximization: near-optimal time complexity meets practical efficiency. In *SIGMOD*, 2014.
- [25] H. Vahabi, I. K. F. Gullo, and M. Halkidi. Difrec: A social-diffusion-aware recommender system. In *CIKM*, 2015.
- [26] Z. Wen and C.-Y. Lin. On the quality of inferring interests from social neighbors. In *KDD*, 2010.
- [27] J. Yang and J. Leskovec. Modeling information diffusion in implicit networks. In *ICDM*, 2010.