Analysis of Influence Contribution in Social Advertising

Yuqing Zhu
School of Computer Science and Engineering
Nanyang Technological University
yuqing002@e.ntu.edu.sg

Jing Tang*
Data Science and Analytics Thrust
The Hong Kong University of Science and Technology
jingtang@ust.hk

Xueyan Tang
School of Computer Science and Engineering
Nanyang Technological University
asxytang@ntu.edu.sg

Lei Chen
Data Science and Analytics Thrust
The Hong Kong University of Science and Technology
leichen@ust.hk

ABSTRACT

Online Social Network (OSN) providers usually conduct advertising campaigns by inserting social ads into promoted posts. Whenever a user engages in a promoted ad, she may further propagate the promoted ad to her followers recursively and the propagation process is known as the *word-of-mouth* effect. In order to spread the promotion cascade widely and efficiently, the OSN provider often tends to select the influencers, who normally have large audiences over the social network, to initiate the advertising campaign. This marketing model, also termed as influencer marketing, has been gaining increasing traction and investment and is rapidly becoming one of the most widely-used channels in digital marketing.

In this paper, we formulate the problem for the OSN provider to derive the influence contributions of influencers given the campaign result, considering the viral propagation of the ads, namely *influence contribution allocation (ICA)*. We make a connection between ICA and the concept of Shapley value in cooperative game theory to reveal the rationale behind ICA. A naive method to obtain the solution to ICA is to enumerate all possible cascades delivering the campaign result, resulting in an exponential number of potential cascades, which is computationally intractable. Moreover, generating a cascade producing the exact campaign result is non-trivial. Facing the challenges, we develop an exact solution in linear time under the linear threshold (LT) model, and devise a *fully polynomial-time randomized approximation scheme (FPRAS)* under the independent cascade (IC) model. Specifically, under the IC model, we propose an efficient approach to estimate the expected influence contribution in probabilistic graphs modeling OSNs by designing a scalable sampling method with provable accuracy guarantees. We conduct extensive experiments and show that our algorithms yield solutions with remarkably higher quality over several baselines and improve the sampling efficiency significantly.

PVLDB Reference Format:
Yuqing Zhu, Jing Tang, Xueyan Tang, and Lei Chen. Analysis of Influence Contribution in Social Advertising. PVLDB, 15(2): 348–360, 2022.
doi:10.14778/3489496.3489514

1 INTRODUCTION

Online Social Networks (OSNs), such as Facebook, Orkut and Twitter, serve as important media where users gain information in the modern world. With the vast number of active users sharing information on social media, OSN providers often turn to utilizing the social connections between users for social advertising. The rich connections serve as fertile soil for advertising campaigns as information can be propagated efficiently and widely with the *word-of-mouth* effects. In an advertising campaign, users actively engage in a promoted ad through social actions such as "like", "share" or "comment". The influencers, who have large audiences over the social network, are usually selected by the OSN provider to initiate the advertising campaign. In incentivized social advertising, the advertiser pays to the OSN provider a cost based on the number of engagements that the influencers bring. Furthermore, a cut of the revenue collected by the OSN providers may in turn be allocated to the influencers for their endorsements of the advertising campaign. Consequently, given the campaign result, to fairly allocate the advertising revenue among the influencers, a natural problem arises that *how much contribution does each influencer make?*

In the existing literature, e.g., in classic influence maximization [25], a set of k influencers are selected to maximize their total influence without considering the individual contributions. In the context of *incentivized social advertising* [2, 3], the OSN providers often use the expected number of engagements that an influencer can bring or the expected number of users activated by the diffusion process, also known as *influence spread*, to compute the seeding cost that the advertiser needs to pay for initiating the campaign. Influence spread, representing the influence capability of influencers, is a possible reference to measuring their contributions in advertising campaigns. However, influence spread may not be an effective indicator of an influencer's contribution when given the set of users engaged in a particular campaign.

In the following, we illustrate the difference between influence contribution studied in this paper and other influence-based metrics in the existing literature, e.g., influence spread. Figure 1 shows a simple network consisting of nodes (users) and directed edges (connections) between nodes. Each edge is associated with a probability that one node successfully influences the other, e.g., u influences w with probability 0.4. Let $\{u, v\}$ be the influencer set. We consider the widely adopted independent cascade (IC) model. The influence spreads of u and v are 1.5 and 1.6. As the influencer set is initially set

*Corresponding author: Jing Tang.

Proceedings of the VLDB Endowment, Vol. 15, No. 2 ISSN 2150-8097.
doi:10.14778/3489496.3489514

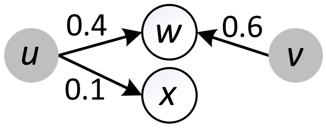


Figure 1: An example network.

active by the OSN provider, we focus on the *influence power*, measured upon the engagements of the users activated by the influencer set without considering the influence spread of the influencers on themselves. It is easy to see that the influence powers of u and v are 0.5 and 0.6 respectively. However, given that w and x are influenced in the advertising campaign, the influence powers of u and v on w should be normalized by a probability of $1-(1-0.4)\cdot(1-0.6)=0.76$ representing that w is activated by at least one influencer in the cascade. Similarly, the influence power of u on x should be normalized by 0.1. Then, the contributions should be $0.4/0.76+0.1/0.1=1.53$ and $0.6/0.76=0.79$ for u and v respectively. In a word, the influence contribution of each influencer should be measured based on its conditional influence power given the observed campaign result.

Moreover, the influence contributions of the influencers expressed by their conditional influence powers given the campaign result may still be imprecise when different influencers can have influence on the same set of users. In the example of Figure 1, u and v can activate w simultaneously while w's engagement is counted only once in the campaign result. In this case, the contribution of u (resp. v) on w is compromised when v (resp. u) successfully activates w and they share the same contribution to activate w. Thus, we need to further split the probability of $0.4\cdot 0.6$ equally between u and v to reflect their true contributions when both u and v activate w. In other words, u's contribution on w is calculated as $\big(0.4\cdot(1-0.6)+(0.4\cdot 0.6)/2\big)=0.28$ and v's contribution on w is $\big(0.6\cdot(1-0.4)+(0.4\cdot 0.6)/2\big)=0.48$. As a result, the influence contributions are $0.28/0.76+1=1.37$ and $0.48/0.76=0.63$ for u and v respectively. Hence, we also need to take into account the cases when a node is influenced by multiple influencers to further refine their contributions given the campaign result.

As illustrated by the above analysis, the influence contributions for u and v given the campaign result are measured upon the conditional influence power and precisely refined to tackle the influence overlap between influencers. The influence contributions of 1.37 and 0.63 for u and v are significantly different from the results of 1.5 and 1.6 by the influence spread method or the results of 0.5 and 0.6 by the influence power method. In this paper, we propose a solution to compute the contributions of the influencers given the result of an advertising marketing campaign. The influence power is hard to obtain as there are exponentially many potential cascades in terms of the number of edges that can deliver the given campaign result. It is computationally intractable to naively enumerate these possible cascades to obtain the influence contribution. A possible alternative is to approximate the exact solutions with acceptable errors by sampling-based methods. Nevertheless, the cascades producing the exact campaign result are hard to sample. The existing sampling methods are ineffective in generating the cascades that produce the given campaign result in our problem and may waste significant computation time on irrelevant cascades.

To obtain the influence contribution, we make the following major contributions in this paper.

- We formally define the contributions of the influencers in a deterministic cascade under two most commonly used diffusion models, including the independent cascade (IC) model and the linear threshold (LT) model. We then propose a novel *influence contribution allocation* (ICA) problem that aims to minimize the average mean-squared-error of influence contribution with respect to every possible cascade delivering the observed campaign result.
- We make a connection between the influence contribution with the concept of Shapley value in cooperative game theory. In particular, we find that the Shapley value is the exact solution for the ICA problem, which confirms the rationality of our definition of influence contribution.
- Under the LT model, a linear time algorithm is proposed to find the exact solution for the ICA problem, which utilizes an elegant bottom-up approach.
- Under the IC model, a fully polynomial-time randomized approximation scheme (FPRAS) is developed. Specifically, we propose a cascade generation method called *influencer backtrack* to significantly enhance the sampling efficiency. We further boost the sampling efficiency of the state-of-the-art stopping rule algorithm by deriving tighter bounds for the influence contributions to estimate while ensuring the accuracy guarantee of our estimation results.
- We perform extensive experiments on real-world datasets with up to millions of nodes and demonstrate the efficacy and efficiency of our algorithms.

Applications. As we mentioned previously, in incentivized social advertising, when the OSN provider allocates a cut of the total revenue to the influencers for their endorsements given the result of an advertising campaign, the ratio measured by the influence contributions provides a *fair* way for allocation.

In identifying the source of misinformation, given the spread result of misinformation and the set of suspectors, we can use the influence contributions of the suspectors as a reference to investigate the most suspected sources of misinformation. That is, the larger influence contribution a suspector provides, the higher probability the supector will be the source.

Different from the existing metrics of influence spread and influence power, we need to consider the influence contributions given the result of the marketing campaign in the above scenarios, i.e., our sample space only consists of the possible cascades where the given set of nodes are activated.

Organization. The rest of this paper is organized as follows. Section 2 introduces the preliminaries and formally defines the problem of influence contribution allocation (ICA). Section 3 analyzes the properties of ICA, especially leveraging the concept of Shapley value. Section 4 presents our proposed solution to the ICA problem under both the IC and LT models. Section 5 reviews the related work. Section 6 discusses the experimental evaluation. Finally, Section 7 concludes the paper.

2 PRELIMINARIES AND PROBLEM DEFINITION

2.1 Social Influence

2.1.1 Diffusion Model. We model an online social network as a directed probabilistic graph $G = (V, E)$ where V are the users and E are the connections among users. Each edge $(u, v) \in E$ is associated with a probability $p_{u,v}$ representing the probability that u can successfully activate v. We denote N_v as the set of node v's neighbors and I_v as the set of v's inverse neighbors, i.e., $N_v := \{w \colon w \in V, (v, w) \in E\}$, and $I_v := \{u \colon u \in V, (u, v) \in E\}$. A campaign starts with a set of influencer nodes $S \subset V$ and follows a diffusion process. We focus on two basic and widely adopted diffusion models, i.e., the *independent cascade (IC)* and *linear threshold (LT)* models [25]. Initially, at timestamp 0, the influencer nodes in S are *activated*, while all the other nodes are *inactive*. When a node first becomes activated at timestamp i, it has a *single* chance to activate its inactive neighbors at timestamp $i + 1$. The active nodes remain active until the end of the diffusion process. The process terminates when no more nodes in the graph can be activated. The difference between the IC and LT models lies in the details of node activation:

- *IC model.* When a node u first becomes activated at timestamp i, it attempts to activate each inactive neighbor v with probability $p_{u,v}$ at timestamp $i + 1$.
- *LT model.* The probabilities of the incoming edges to each node v from its inverse neighbors I_v satisfy $\sum_{u \in I_v} p_{u,v} \leq 1$. Each node v randomly selects a threshold $\lambda_v \in [0, 1]$. If a node v is inactive at timestamp i, v becomes activated at timestamp $i + 1$ only if $\sum_{u \in A(I_v)} p_{u,v} \geq \lambda_v$, where $A(I_v) \subseteq I_v$ is the set of v's inverse neighbors that are activated at timestamp i.

2.1.2 Influence Spread. Due to the stochastic nature of the diffusion process, a campaign starting from an influencer set S can generate many different cascades. The *influence spread* of an influencer set S, referred to as $\sigma(S)$, is the expected number of nodes activated by the diffusion process starting from S over all possible cascades. Furthermore, we denote by $\sigma_\omega(S)$ the influence spread of S under a given cascade ω. Usually, the influencer set is initially set active by the OSN provider, i.e., their self-engagements are initialized by the OSN provider and should not be counted as their contributions. Thus, we focus on the number of engagements that S can bring, which excludes the self-influence artifact in influence spread. We define this new influence measure as *influence power*, namely $\sigma^*(S)$, i.e.,

$$\sigma^*(S) := \sigma(S) - |S|.$$

2.2 Problem Definition

2.2.1 Influence Contribution. On the basis of influence power, understanding the individual contribution of each influencer in S is crucial. For instance, in the context of incentivized social advertising [2, 3], influence contribution can be used as the indicator for fair revenue allocation. Given a specific cascade ω, we formally define the influence contribution as follows.

Under the IC model, we say that a node v is influenced by an influencer s if there is an activation path from s to v. Note that an activated node v might be influenced by multiple influencers, as there might be several inverse neighbors of v activating v simultaneously at timestamp $i + 1$. Note also that each inverse neighbor u of v that first becomes activated at timestamp i attempts to activate v *independently* at timestamp $i + 1$. Therefore, to measure the contribution fairly with respect to the independence on activation, we assign each influencer that influences the same node with an equal share of contribution. The influence contribution of each influencer under the IC model is formally defined as follows.

Definition 2.1. Under the IC model, given an influencer set S and a cascade ω, denote by $T_\omega(s)$ the set of nodes influenced by s in the cascade ω for each $s \in S$, and by $S_\omega(v)$ the set of influencers influencing v for each $v \in T_\omega(S)$. Then, for each influencer $s \in S$, its influence contribution, namely $\psi_\omega(s)$, is given by

$$\psi_\omega(s) := \sum_{v \in T_\omega(s)} \frac{1}{|S_\omega(v)|}. \tag{1}$$

Under the LT model, when a node v first becomes activated at timestamp $i + 1$, all the active inverse neighbors of v at timestamp i contribute to the activation of v. Different from the IC model, such an activation is due to the aggregate probability weight, which indicates that the node u with larger probability weight $p_{u,v}$ shall contribute more. Therefore, to distribute the contribution in a fair manner, we consider that u's contribution on v is proportional to $p_{u,v}$. The influence contribution under the LT model is formally defined as follows.

Definition 2.2. Under the LT model, given an influencer set S and a cascade ω, denote by N_v^ω be the subset of neighbors of v that become activated after v, and by I_v^ω be the subset of inverse neighbors of v that become activated before v. Then, the influence contribution $\psi_\omega(u)$ of each activated node u is given by

$$\psi_\omega(u) := \sum_{v \in N_u^\omega} \left(\frac{p_{u,v}}{\sum_{w \in I_v^\omega} p_{w,v}} \cdot \big(1 + \psi_\omega(v)\big) \right). \tag{2}$$

In the above discussion, we assume that the true cascade is known. However, in real applications, observing the true cascade, i.e., which node activates which node or the *edge* status, is hard. Instead, we can easily obtain the activation status of each node, i.e., the *node* status. In this paper, we focus on the node level feedback model. Formally, given a set of influencers S, let $\mathcal{T} = (T(S), t)$ be the node status observed from the influence propagation, where $T(S)$ is the set of non-influencer nodes activated by S and $t(v)$ indicates the activation timestamp of v for each $v \in T(S)$. Given a node level feedback $\mathcal{T}$, there may be several cascades that can produce the same result. For such a cascade ω, we write $\omega \sim \mathcal{T}$. Given a cascade ω, we denote $\mathcal{T}_\omega$ as the node status observed in the cascade ω. Let $\psi_{\mathcal{T}_\omega}(s)$ be the influence contribution derived for the influencer s based on the observation $\mathcal{T}_\omega$. We measure the quality based on square-error, i.e., $(\psi_{\mathcal{T}_\omega}(s) - \psi_\omega(s))^2$. The goal of the *influence contribution allocation (ICA)* problem is to find the influence contribution of each influencer s based on the observed node status such that the average *mean-squared-error (MSE)* for all influencers is minimized, i.e.,

$$\operatorname*{arg\,min}_{\forall s, \omega \colon \psi_{\mathcal{T}_\omega}(s)} \frac{1}{|S|} \sum_{s \in S} \mathbb{E}_\omega \Big[\big(\psi_{\mathcal{T}_\omega}(s) - \psi_\omega(s)\big)^2 \Big], \tag{3}$$

where the expectation is taken over the randomness of cascade ω. The ICA problem can be decomposed into an elementary version.

Definition 2.3. Given an observed $\mathcal{T}$, the ICA problem aims to find a *minimum mean-square-error (MMSE)* estimator $\psi(s)$ of influence contribution for each influencer $s \in S$, i.e.,

$$\psi(s) := \arg\min_{\psi} \mathbb{E}_{\omega \sim \mathcal{T}}[(\psi - \psi_\omega(s))^2], \tag{4}$$

where the expectation is taken over the randomness of ω with respect to $\omega \sim \mathcal{T}$.

Note that (4) is equivalent to the following problem.

$$\arg\min_{\forall s,\omega:\, \psi_{\mathcal{T}_\omega}(s)} \sum_{\omega \sim \mathcal{T}} \Pr[\omega \sim \mathcal{T}] \cdot \left(\psi^2_{\mathcal{T}_\omega}(s) - 2\psi_{\mathcal{T}_\omega}(s) \cdot \psi_\omega(s) + \psi^2_\omega(s)\right).$$

It is trivial to verify that $\psi(s)$ obtained from (4) for each $s \in S$ and each possible $\mathcal{T}$ is the optimal solution to the ICA optimization problem given in (3). In the rest of the paper, we will focus on the elementary version of the ICA problem given in Definition 2.3.

2.3 Shapley Value

The Shapley value is a solution concept known as natural interpretations of contribution in cooperative game theory. Solution concepts are related to how much the value of a coalition is distributed to each player in the game. In game theory, a game is defined by the following components: a set of players, a set of possible actions that the player can take and a utility function that relates an action to a payoff. A cooperative or coalitional game is where the players can coalesce to achieve higher utilities. Formally, given a set S of k players $\{s_1, s_2, \ldots, s_k\}$, a cooperative game is defined as a pair (S, f), where f is a characteristic function expressing the worth or value of a coalition, i.e., f maps each subset of S to a (non-negative) real number, $f\colon 2^{|N|} \mapsto \mathbb{R}_+$. A solution concept for S is a vector $\vec{x} = \langle x_1, \ldots, x_k \rangle$, where x_i expresses the value that should be allocated to the player s. Among all solution concepts, the Shapley value [34] is often used in the value-sharing literature to prescribe the fair amount that a player should receive when it shares a coalition with a set of players. Note that k players can form a coalition in $k!$ different ways assuming all the orders that the players participate in the coalition are equally possible. In each permutation of the k players, when a player joins the coalition, it makes a marginal contribution given the existing players. The Shapley value of a player is the average marginal contribution it makes to all the permutations. Denote Π as the set of all $k!$ permutations, $\pi \in \Pi$ as a possible permutation, and $S_{s,\pi}$ as the set of players joining the coalition before player s in π. Then, the Shapley value $\phi_f(s)$ of each player s in the game (S, f) is given by

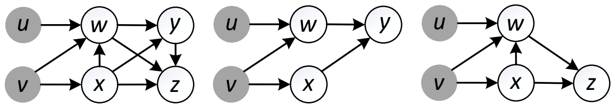
$$\phi_f(s) = \frac{1}{k!} \sum_{\pi \in \Pi} \left(f(S_{s,\pi} \cup \{s\}) - f(S_{s,\pi})\right). \tag{5}$$

Computing the Shapley value for a general cooperative game requires intensive calculation and it is known to be #P-hard [15].

The Shapley value provides the following properties [34]:

(1) **Efficiency**: $\sum_{s \in S} \phi_f(s) = f(S)$, i.e., the sum of the Shapley values of all players is equal to the value of the grand coalition.

(2) **Symmetry**: For any players $s_i, s_j \in S$, if $\forall S' \subseteq S \setminus \{s_i, s_j\}$, $f(S' \cup \{s_i\}) = f(S' \cup \{s_j\})$, then $\phi_f(s_i) = \phi_f(s_j)$, i.e., if two players have the same marginal contributions on any coalition, they have the same Shapley values.



(a) Graph G (b) Cascade graph (IC) (c) Cascade graph (LT)

Figure 2: Example cascade graph.

(3) **Linearity**: Given two characteristic functions f_1 and f_2, for any $x, y > 0$ and $S' \subseteq S$, $\phi_{xf_1+yf_2}(S') = x \cdot \phi_{f_1}(S') + y \cdot \phi_{f_2}(S')$, i.e., the Shapley values of a linear combination of two characteristic functions are given by the linear combination of the Shapley values of the individual characteristic functions.

(4) **Null Player**: For any player $s \in S$, if $\forall S' \subseteq S \setminus \{s\}$, $f(S' \cup \{s\}) - f(S') = 0$, then $\phi_f(s) = 0$, i.e., if a player has zero marginal contribution on any coalition, its Shapley value is 0.

3 PROPERTY OF INFLUENCE CONTRIBUTION

In this section, we give several important properties of influence contributions, especially leveraging the concept of Shapley value.

3.1 MMSE Estimator

We first show that the MMSE estimator $\psi(s)$ of influence contribution defined in (4) is the conditional expectation of $\psi_\omega(s)$ given the known observed activation status $\mathcal{T}$.

Proposition 3.1. *Given an observed $\mathcal{T}$, $\psi(s)$ satisfies*

$$\psi(s) = \mathbb{E}_{\omega \sim \mathcal{T}}[\psi_\omega(s)]. \tag{6}$$

Proof. Given a $\mathcal{T}$, we have

$$\begin{aligned}
&\mathbb{E}_{\omega \sim \mathcal{T}}[(\psi - \psi_\omega(s))^2] \\
&= \psi^2 - 2\psi \mathbb{E}_{\omega \sim \mathcal{T}}[\psi_\omega(s)] + \mathbb{E}_{\omega \sim \mathcal{T}}[(\psi_\omega(s))^2] \\
&= \Big(\psi - \mathbb{E}_{\omega \sim \mathcal{T}}[\psi_\omega(s)]\Big)^2 + \mathbb{E}_{\omega \sim \mathcal{T}}[(\psi_\omega(s))^2] - \mathbb{E}^2_{\omega \sim \mathcal{T}}[\psi_\omega(s)],
\end{aligned}$$

which is minimized when $\psi = \mathbb{E}_{\omega \sim \mathcal{T}}[\psi_\omega(s)]$. □

3.2 Cascade Graph

Given an influencer set S and an observed node activation status $\mathcal{T}$, we extract a subgraph D from the original graph G, referred to as *cascade graph*, that characterizes the potential cascades. Specifically, all the nodes in $S \cup T(S)$ form the node set of D. The edge sets of D under the IC model and the LT model are slightly different.

- **IC Model.** For every $(u, v) \in G$, $(u, v) \in D$ if $t(u) = t(v) - 1$.
- **LT Model.** For every $(u, v) \in G$, $(u, v) \in D$ if $t(u) < t(v)$.

Example 3.2. As shown in Figure 2, given a social graph in Figure 2(a) with campaign result $\mathcal{T} = \big(\{w, x, y\}, \{t(x) = t(w) = 1, t(y) = 2\}\big)$ under the IC model, the cascade graph D extracted from G is shown in Figure 2(b). Given the campaign result $\mathcal{T} = \big(\{w, x, z\}, \{t(x) = 1, t(w) = 2, t(z) = 3\}\big)$ under the LT model, the cascade graph D extracted from G is shown in Figure 2(c).

Clearly, every possible cascade $\omega \sim \mathcal{T}$ is based on D. Thus, unless specified otherwise, our discussions are on D only and the notations are also on D, e.g., I_v is the set of v's inverse neighbors in D. It is trivial to see that D is a directed acyclic graph (DAG).

Proposition 3.3. *D is a directed acyclic graph.*

Proof. If there is a cycle consisting of nodes $v_1, v_2, \ldots, v_i$ in D, we have $t(v_1) < t(v_2) < \cdots < t(v_i) < t(v_1)$, which shows a contradiction. This completes the proof. □

Proposition 3.3 shows that we can avoid generating cycles when tracing back to the influencer influencing the node in $T(S)$ and leads to Proposition 3.4, which is the key to our design of the influencer backtrack sampling method to efficiently generate only the relevant cascades as shall be presented in Section 4.3.

Kempe et al. [25] showed that the diffusion process can be described by a live/blocked edge approach. In a nutshell, for each node v, the edges from I_v being live follow the subset sampling under the IC model and the proportional sampling under the LT model, respectively. Specifically, under the IC model, for each edge (u, v), we can flip a biased coin to set the edge live with probability $p_{u,v}$ and set it blocked with probability $1 - p_{u,v}$. Under the LT model, for each node v, with probability $1 - \sum_{w \in I_v} p_{w,v}$, all the edges from I_v to v are blocked, and with probability $\sum_{w \in I_v} p_{w,v}$, an edge from one inverse neighbor $u \in I_v$ to v is set to live following the distribution $p_{u,v} / \sum_{w \in I_v} p_{w,v}$ while the other edges to v are blocked. Removing all the blocked edges from D, we can obtain a *realization g* of D with live edges only, and write $g \sim D$. The nodes that are reachable from S under the realization g are activated, and the other nodes are not activated. We say that g is relevant, namely $g \sim \mathcal{T}$, if every node in $T(S)$ is reachable from S under the realization g. We have the following proposition for the relevant realizations.

Proposition 3.4. *A necessary and sufficient condition for $g \sim \mathcal{T}$ is that under the realization g, each node in $T(S)$ must have at least one live incoming edge from its inverse neighbors.*

Proof. (Necessary) On one hand, if a node $v \in T(S)$ does not have any live incoming edge from its inverse neighbors in a realization, v would be inactive in the campaign result and this realization would be irrelevant.

(Sufficient) On the other hand, consider that every node $v \in T(S)$ has a live incoming edge. For each $v \in T(S)$, find the longest path that ends at v in the realization consisting of nodes $v_i, v_{i-1}, \ldots, v_1, v$. If $v_i \in T(S)$, since D is a DAG, there must exist a live edge (v_j, v_i) such that v_j does not belong to this path, which contradicts to the definition of the longest path. Hence, $v_i \in S$, which indicates that v will be activated. □

For any relevant realization g, we further establish the relation between $\Pr[g \sim \mathcal{T}]$ and $\Pr[g \sim D]$.

Proposition 3.5. *For each relevant realization $g \sim \mathcal{T}$, we have*

$$\Pr[g \sim \mathcal{T}] = \frac{1}{\prod_{v \in T(S)} \beta_v} \cdot \Pr[g \sim D], \tag{7}$$

where β_v is the probability that v has at least one live incoming edge from its inverse neighbors in D sampling from the live/blocked edge approach.

Proof. Following Proposition 3.4, for any relevant realization $g \sim \mathcal{T}$, each node $v \in T(S)$ needs to have at least one live incoming edge from its inverse neighbors in D. As such events for each node $v \in T(S)$ are independent, the probability for all these events to happen simultaneously is given by $\prod_{v \in T(S)} \beta_v$, i.e.,

$$\sum_{g \sim \mathcal{T}} \Pr[g \sim D] = \Big(\prod_{v \in T(S)} \beta_v \Big) \cdot \sum_{g \sim \mathcal{T}} \Pr[g \sim \mathcal{T}].$$

Meanwhile, for two cascades $g, g' \sim \mathcal{T}$, it holds that

$$\frac{\Pr[g \sim D]}{\Pr[g' \sim D]} = \frac{\Pr[g \sim \mathcal{T}]}{\Pr[g' \sim \mathcal{T}]}.$$

Putting it together completes the proof. □

3.3 An Alternative Definition for LT Model

For any set S', we denote by $\sigma^*_g(S')$ the number of nodes that are reachable from S' but are not in S', i.e., influence power of S', under the realization g, and by $\hat{\sigma}^*(S')$ the expectation of $\sigma^*_g(S')$ with respect to $g \sim \mathcal{T}$, i.e.,

$$\hat{\sigma}^*(S') := \mathbb{E}_{g \sim \mathcal{T}}[\sigma^*_g(S')]. \tag{8}$$

In a more intuitive way, the influence contribution under the LT model can also be measured in a similar way to that under the IC model given a cascade ω (Definition 2.1), i.e.,

$$\psi_\omega(s) := \sum_{v \in T_\omega(s)} \frac{1}{|S_\omega(v)|},$$

where $T_\omega(s)$ is the set of nodes influenced by s in the cascade ω for each $s \in S$ and $S_\omega(v)$ is the set of influencers influencing v for each $v \in T_\omega(s)$. Under the LT model, there is exactly one live incoming edge for each $v \in T(S)$ by the live/blocked edge approach, so we have $|S_\omega(v)| = 1$, i.e.,

$$\psi_\omega(s) = |T_\omega(s)|.$$

Based on (6), for any possible realization $g \sim \mathcal{T}$, we have

$$\psi(s) = \mathbb{E}_{g \sim \mathcal{T}}[\psi_\omega(s)] = \mathbb{E}_{g \sim \mathcal{T}}[|T_\omega(s)|] = \hat{\sigma}^*(s). \tag{9}$$

Then, for any $u \in T(S)$, we have

$$\begin{aligned} \psi(u) &= \sum_{v \in N_u} \big(\mathbf{1}_{\{(u,v) \in g\}} \cdot (1 + \hat{\sigma}^*(v)) \big) \\ &= \sum_{v \in N_u} \big(\mathbf{1}_{\{(u,v) \in g\}} \cdot (1 + \psi(v)) \big). \end{aligned}$$

Note that $\Pr[(u, v) \in g] = \frac{p_{u,v}}{\beta_v}$ where $\beta_v = \sum_{u \in I_v} p_{u,v}$. Thus, we have

$$\psi(u) = \sum_{v \in N_u} \big(\frac{p_{u,v}}{\beta_v} \cdot (1 + \psi(v)) \big). \tag{10}$$

As a result, the above measurement of influence contribution also yields the same result as Definition 2.2, which validates the rationale of our influence contribution definition under the LT model.

3.4 Connection to Shapley Value

We show that the Shapley value in the cooperative game $\big(S, \hat{\sigma}^*(\cdot)\big)$ is exactly the MMSE estimator of influence contribution.

Theorem 3.6. *For each influencer $s \in S$, let $\phi(s)$ be the Shapley value of s in the cooperative game $\big(S, \hat{\sigma}^*(\cdot)\big)$. Then, $\phi(s) = \psi(s)$.*

Proof. By definition, the Shapley value $\phi(s)$ of s is given by

$$\begin{aligned}
\phi(s) &= \frac{1}{|S|!} \cdot \sum_{\pi \in \Pi} \Big(\hat{\sigma}^*(S_{s,\pi} \cup \{s\}) - \hat{\sigma}^*(S_{s,\pi})\Big) \\
&= \frac{1}{|S|!} \cdot \mathbb{E}_{g \sim \mathcal{T}}\Big[\sum_{\pi \in \Pi} \Big(\hat{\sigma}_g(S_{s,\pi} \cup \{s\}) - \hat{\sigma}_g(S_{s,\pi})\Big)\Big] \\
&= \frac{1}{|S|!} \cdot \mathbb{E}_{g \sim \mathcal{T}}\Big[\sum_{\pi \in \Pi} \Big(|T_g(S_{s,\pi} \cup \{s\})| - |T_g(S_{s,\pi})|\Big)\Big] \\
&= \frac{1}{|S|!} \cdot \mathbb{E}_{g \sim \mathcal{T}}\Big[\sum_{\pi \in \Pi} \Big(|T_g(\{s\}) \setminus T_g(S_{s,\pi})|\Big)\Big],
\end{aligned}$$

where $T_g(S')$ is the set of nodes reachable from S' but excluding S' under g. For any $v \in T_g(\{s\})$, if $v \notin T_g(S_{s,\pi})$, then in the permutation π, s is placed the first among all the influencers in $S_g(v)$ that can reach v under g. Meanwhile, for each $v \in T_g(\{s\})$, among the set Π of all $|S|!$ permutations, there are a fraction $\frac{1}{|S_g(v)|}$ of permutations such that s is placed before other influencers in $S_g(v)$. Hence,

$$\begin{aligned}
\sum_{\pi \in \Pi} \Big(|T_g(\{s\}) \setminus T_g(S_{s,\pi})|\Big) &= \sum_{\pi \in \Pi} \sum_{v \in T_g(\{s\})} \mathbf{1}_{\{v \notin T_g(S_{s,\pi})\}} \\
&= \sum_{v \in T_g(\{s\})} \sum_{\pi \in \Pi} \mathbf{1}_{\{v \notin T_g(S_{s,\pi})\}} \\
&= \sum_{v \in T_g(\{s\})} \frac{|S|!}{|S_g(v)|},
\end{aligned}$$

where $\mathbf{1}_{\{v \notin T_g(S_{s,\pi})\}}$ is an indicator function such that $\mathbf{1}_{\{v \notin T_g(S_{s,\pi})\}} = 1$ if $v \notin T_g(S_{s,\pi})$ and $\mathbf{1}_{\{v \notin T_g(S_{s,\pi})\}} = 0$ otherwise. As a result, we have

$$\phi(s) = \mathbb{E}_{g \sim \mathcal{T}}\Big[\sum_{v \in T_g(\{s\})} \frac{1}{|S_g(v)|}\Big]. \tag{11}$$

Observe that (i) $(u, v) \in g$ only if $(u, v) \in G$ and $t(u) = t(v) - 1$ under the IC model or $t(u) < t(v)$ under the LT model, and (ii) an independent test is taken to decide whether (u, v) is live (with probability of $p_{u,v}$). As can be seen, the distribution of $g \sim \mathcal{T}$ is exactly the same as that of $\omega \sim \mathcal{T}$. Hence, putting it all together of (1), (6) and (11) yields that $\phi(s) = \psi(s)$ under the IC model. Meanwhile, as we have shown in Section 3.3, (11) also leads to Definition 2.2 and thus indicates that $\phi(s) = \psi(s)$ under the LT model. This completes the proof. □

Theorem 3.6 states that our MMSE estimator of influence contribution is the Shapley value. Recall from Section 2.3 that, the Shapley value provides the properties of *efficiency*, *symmetry*, *linearity* and *null player*, which also apply to our MMSE estimator of influence contribution. These properties again confirm that the contribution of each influencer is indeed well characterized by our solution.

Algorithm 1: Influence Contribution under LT

```
  Input: influencer set $S$, cascade graph $D$, and observation $\mathcal{T}$
  Output: $\psi(s)$ for each influencer $s \in S$
1 foreach node $v \in T(S)$ do
2     $\beta_v \leftarrow \sum_{u \in I_v} p_{u,v}$;
3 for $j \leftarrow$ stopping timestamp to 0 do
4     foreach node $u$ with $t(u) = j$ do
5         $\psi(u) \leftarrow \sum_{v \in N_u} \Big(\frac{p_{u,v}}{\beta_v} \cdot \big(1 + \psi(v)\big)\Big)$;
6 return $\{\psi(s) \colon s \in S\}$;
```

3.5 Discussion on Continuous Activation Time

In practice, the activation time $t(v)$ of each node v is a real number, rather than the discrete timestamp. Then, under the IC model, a node is unlikely activated by two or more nodes simultaneously, since different nodes should activate the same node at different times when the time is a real number. Hence, for each influenced node $v \in T(S)$, there is a unique influencer $s \in S$ that influences v. Note that upon observing the activation time of each node, we still do not know the actual cascade. For each $v \in T(S)$, it can be activated by any inverse neighbor u in G that has an activation time earlier than v, i.e., $u \in I_v$ and $t(u) < t(v)$. Then, the cascade graph construction needs a slight modification. That is, we keep $(u, v) \in D$, if $(u, v) \in G$ and $t(u) < t(v)$. We demonstrate in the experiments that our proposed solution still performs rather well in terms of minimizing the square-error. Interestingly, under the LT model, there is no difference no matter whether the activation time is a real number or discrete value. The reason is when a node is activated, all the nodes with activation times earlier than its activation time will share the contribution proportional to the probability weight.

4 EFFICIENT COMPUTATION

In this section, we present a linear time algorithm for computing the exact influence contribution under the LT model, while under the IC model, we devise a *fully polynomial-time randomized approximation scheme (FPRAS)* that can efficiently and accurately estimate the true influence contribution leveraging the notion of Shapley value.

4.1 Exact Solution for LT Model

Based on Definition 2.2, we can calculate the influence contributions under the LT model using a bottom-up approach. Specifically, given an observed activation status $\mathcal{T}$, if a node u does not activate any node, i.e., u has no neighbor in the cascade graph D, by definition, we have $\psi(u) = 0$. Suppose that the diffusion process stops after timestamp i. Then, we can sequentially compute the influence contributions for the nodes with the activation timestamps of $i - 1, i - 2, \ldots, 1, 0$.

Algorithm 1 shows the pseudo code of the calculation of $\psi(s)$ for each $s \in S$. We first compute β_v for each $v \in T(S)$ (Lines 1–2). Then, we calculate $\psi(u)$ using a bottom-up approach (Lines 3–5). Note that if $N_u = \emptyset$ in the cascade graph D, we set $\psi(u) = 0$ (Line 5).

Example 4.1. We use a simple example to illustrate the procedure of Algorithm 1. Figure 3 shows a cascade graph D where $S = \{s_1, s_2\}$ and $T(S) = \{u, v, w, x, y\}$. Edges (s_1, u) and (s_2, w) have normalized propagation probabilities of 1. Suppose that other edges (u', v')

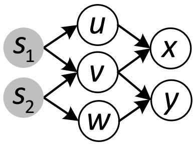


Figure 3: A cascade graph D.

have normalized propagation probabilities of $\frac{p_{u',v'}}{\beta_{v'}} = 0.5$. First, we compute $\psi(x) = \psi(y) = 0$. Then, we compute $\psi(u) = \psi(w) = 0.5 \times (1+0) = 0.5$ and $\psi(v) = 0.5 + 0.5 = 1$. Hence, $\psi(s_1) = \psi(s_2) = 1 \times (1+0.5) + 0.5 \times (1+1) = 2.5$.

Time Complexity. It takes $O(\sum_{v\in T(S)} |I_v|) = O(m_D)$ time for computing β_v for all the nodes $T(S)$, where m_D is the number of edges in D. Then, it takes $O(\sum_{u\in S\cup T(S)} |N_u|) = O(m_D)$ time for computing $\psi(u)$ for all the nodes $S \cup T(S)$. Therefore, the total complexity of Algorithm 1 is $O(m_D)$.

4.2 An FPRAS for IC Model

A naive way to compute $\psi(s)$ of each influencer s is to enumerate every possible cascade ω to calculate $\psi_\omega(s)$. Then, by Proposition 3.1, taking the expectation of $\psi_\omega(s)$ gives rise to $\psi(s)$. However, in the cascade graph D, there are m_D edges, resulting in an exponential number $O(2^{m_D})$ of potential cascades which is computationally intractable. More importantly, it is unclear how to generate a cascade that can exactly produce the activation status $\mathcal{T}$. A naive method is to perform an adequate number of Monte Carlo simulations starting from the influencer set S to generate the cascades exactly producing $\mathcal{T}$. Meanwhile, the existing reverse influence sampling technique [5] may also be adopted to generate the cascades. However, the cascades exactly producing $\mathcal{T}$ may not be easily obtained using these methods and a large number of irrelevant cascades would be generated in the sampling process.

Example 4.2. Figure 4 shows an example probabilistic graph consisting of three nodes. Suppose that u is the only influencer selected to start the campaign. Given the campaign result that both v and w are influenced by u, if we perform the Monte Carlo simulations from u, we can have one out of ten simulations in expectation (i.e., $\frac{1}{0.5\cdot 0.2}$) to produce exactly the same result. Similarly, if we adopt the reverse influence sampling technique starting from v and w to produce the cascades, we also expect to see u not activating both v and w in 90% of the cascades. From this simple example, we can infer that, given a campaign result $\mathcal{T}$, most of the cascades generated by the existing Monte Carlo simulation or reverse influence sampling techniques would be irrelevant cascades that do not match the observed result $\mathcal{T}$, wasting a significant amount of computation time.

Facing the challenges, we propose a fully polynomial-time randomized approximation schme (FPRAS), which finds an (ε, δ)-approximation $\tilde{\psi}(s)$ of $\psi(s)$ for each influencer $s \in S$ in polynomial time for any parameters ε and δ, i.e.,

$$\Pr\left[(1-\varepsilon)\psi(s) \le \tilde{\psi}(s) \le (1+\varepsilon)\psi(s)\right] \ge 1-\delta.$$

Our solution consists of two components: (i) we propose *influencer backtrack sampling* to generate a relevant cascade unbiasedly and calculate the influence contribution in the generated cascade efficiently, and (ii) we adopt a stopping rule algorithm to decide the number of relevant cascades that need to be generated to provide the accuracy guarantee. The sampling-based method, generating the relevant cascades efficiently, provides unbiased estimators to well approximate the influence contributions that minimize the average mean-squared-error.

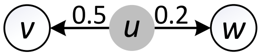


Figure 4: An example graph.

4.3 Influencer Backtrack Sampling for IC Model

Proposition 3.5 shows that if a realization g is directly generated from D, this realization is irrelevant with a high probability of $1 - \frac{1}{\prod_{v\in T(S)} \beta_v}$, which is highly inefficient. In what follows, we present an algorithm that ensures to generate a relevant realization in an unbiased manner.

For each node $v \in T(S)$, assume a fixed order on the inverse neighbors of v in D as $v_1, v_2, \ldots, v_{\ell_v}$ where $\ell_v = |I_v|$. Let A^v represent the event that v has at least one live incoming edge from its inverse neighbors. Let A^v_i be the event that (v_i, v) is the first live incoming edge to v. By the definition of the IC model, we have

$$\Pr[A^v_i] = p_{v_i,v} \cdot \prod_{j=1}^{i-1} (1 - p_{v_j,v}). \tag{12}$$

Since all the events A^v_i are disjoint, we have

$$\beta_v = \Pr[A^v] = \sum_{i=1}^{\ell_v} \Pr[A^v_i] = 1 - \prod_{i=1}^{\ell_v} (1 - p_{v_i,v}). \tag{13}$$

Hence, the first live incoming edge can be selected according to the probability distribution of $\frac{\Pr[A^v_i]}{\beta_v}$. According to Proposition 3.4, we can select at least one live incoming edge for each node in $T(S)$ to generate a relevant cascade $\omega \sim \mathcal{T}$. After obtaining the first live incoming edge (v_i, v), the conventional sampling procedure is applied to examine the remaining edges, i.e., each edge (v_k, v) is live with a probability of $p_{v_k,v}$ for $k > i$. As a result, the probability of (v_k, v) being live is

$$\begin{aligned}
&\Pr[(v_k, v) \text{ is live}] \\
&= \frac{\Pr[A^v_k]}{\beta_v} + p_{v_k,v} \sum_{i=1}^{k-1} \frac{\Pr[A^v_i]}{\beta_v} \\
&= \frac{p_{v_k,v}}{\beta_v} \cdot \Big(\prod_{j=1}^{k-1} (1 - p_{v_j,v}) + 1 - \prod_{i=1}^{k-1} (1 - p_{v_i,v}) \Big) = \frac{p_{v_k,v}}{\beta_v}.
\end{aligned}$$

Therefore, such a cascade is an unbiased sample.

A naive method to estimate $\psi(s)$ is to (i) generate an unbiased sample of cascade ω, and (ii) construct $S_\omega(v)$, i.e., the set of influencers reversely reachable from v, via a reverse breadth first search (BFS) on ω for each $v \in T(S)$. Note that generating ω takes $O(m_D)$ time and performing a BFS for every $v \in T(S)$ takes $O(|T(S)| \cdot m_D)$. To facilitate the calculation, we directly construct the backtracked influencer sets $S_\omega(v)$ for each node $v \in T(S)$ in the sampling procedure, which only performs one pass of traversal.

Algorithm 2: Influencer Backtrack Sampling

```
Input: influencer set $S$, cascade graph $D$, and observation $\mathcal{T}$
Output: generate a cascade $\omega \sim \mathcal{T}$ and calculate $\psi_\omega(s)$ for
        each $s \in S$
1  initialize $S_\omega(v) \leftarrow \emptyset$ for each node $v \in T(S)$, $S_\omega(s) \leftarrow \{s\}$,
   $\psi_\omega(s) \leftarrow 0$ for each influencer $s \in S$;
2  for $j \leftarrow 1$ to stopping timestamp do
3      foreach node $v$ with $t(v) = j$ do
4          select the first live edge $(v_i, v)$ according to the
             probabilty distribution of $\frac{\Pr[A_i^v]}{\beta_v}$;
5          $S_\omega(v) \leftarrow S_\omega(v) \cup S_\omega(v_i)$;
6          foreach $i < k \le \ell_v$ do
7              with probabilty $p_{v_k,v}$: $S_\omega(v) \leftarrow S_\omega(v) \cup S_\omega(v_k)$;
8  foreach influencer $s \in S_\omega(v)$ do
9      $\psi_\omega(s) \leftarrow \psi_\omega(s) + \frac{1}{|S_\omega(v)|}$;
10 return $\{\psi_\omega(s) : s \in S\}$;
```

Algorithm 2 illustrates the detailed procedure to generate a cascade $\omega \sim \mathcal{T}$ and build the backtracked influencer set $S_\omega(v)$ for each node $v \in T(S)$ (i.e., the set of influencers that can reach v in the cascade ω). Initially, the backtracked influencer set $S_\omega(s)$ for each influencer $s \in S$ includes the influencer itself, while the backtracked influencer sets of the non-influencer nodes $T(S)$ are empty (Line 1). The nodes in $T(S)$ are examined by their activation timestamps in ascending order (Lines 2–3). For each node $v \in T(S)$, we start by selecting the first live incoming edge (v_i, v) (Line 4) and update the backtracked influencer set $S_\omega(v)$ by $S_\omega(v) \cup S_\omega(v_i)$ (Line 5). Each remaining incoming edge (v_k, v) where $k > i$ is set to live independently with probability $p_{v_k,v}$ and we update $S_\omega(v)$ by $S_\omega(v) \cup S_\omega(v_k)$ if (v_k, v) is live (Lines 6–7). Finally, we iterate through the backtracked influencer set $S_\omega(v)$ and assign an additive factor of $1/|S_\omega(v)|$ to each influencer $s \in S_\omega(v)$ (Line 9). The influence contributions of the influencers in cascade ω are then returned (Line 10), which are unbiased estimators of $\psi(\cdot)$'s (alternatively the Shapley values), i.e., for each influencer $s \in S$,

$$\psi(s) = \mathbb{E}[\psi_\omega(s)],$$

where the expectation is taken over the randomness of ω generated.

Time Complexity. The time complexity of the sampling phase to get the live edges is $O(m_D)$. The total number of live edges is $\mathrm{TE} = \sum_{v \in T(S)} \frac{\sum_{u \in I_v} p_{u,v}}{\beta_v}$ in expectation and is $O(m_D)$ in the worst case. When an edge is live, a union operation is performed (Lines 5 and 7), which takes $O(|S|)$ time. Thus, constructing the backtracked influencer sets takes $O(|S| \cdot \mathrm{TE})$ time in expectation and $O(|S| \cdot m_D)$ time in the worst case. Therefore, the total time complexity of Algorithm 2 is $O(|S| \cdot \mathrm{TE} + m_D)$ in expectation and $O(|S| \cdot m_D)$ in the worst case.

Discussion. In the literature of machine learning, the concept of backtracking is adopted to predict the preceding states that terminate at a given state [18]. In addition, it is also applied to trace the features that determine the classification results after the network is trained [13]. In general, the similar idea is adopted in our influencer backtracking to trace the influencers that produce the given campaign result in the network. Interestingly, based on Proposition 3.4, our influencer backtracking generates only the relevant samples to produce the given campaign result, which largely boosts the sampling efficiency. This idea of sampling at least 1 incoming neighbor may also be applied to improve the sampling efficiency in machine learning algorithms.

Algorithm 3: Bounds on Influence Contribution

```
Input: influencer set $S$, cascade graph $D$, and observation $\mathcal{T}$
Output: lower and upper bounds $a_s$ and $b_s$ on each influencer
        $s$'s influence contribution in all possible cascades
1  initialize $S(s) \leftarrow \{s\}$, $a_s \leftarrow 0$, $b_s \leftarrow 0$ for each $s \in S$;
2  for $j \leftarrow 1$ to stopping timestamp do
3      foreach node $v$ with $t(v) = j$ do
4          $S(v) \leftarrow \bigcup_{u \in I_v} S(u)$;
5  foreach node $v \in T(S)$ do
6      foreach influencer $s \in S(v)$ do
7          $b_s \leftarrow b_s + 1$;
8      if $|S(v)| = 1$ then
9          $s \leftarrow$ influencer in $S(v)$;
10         $a_s \leftarrow a_s + 1$;
11 return $\{(a_s, b_s) : s \in S\}$;
```

4.4 Accurate Estimate for IC Model

To obtain the accuracy guarantee under the IC model, we adopt the state-of-the-art stopping rule algorithm [47] to estimate the expected influence contributions of all the influencers simultaneously. The stopping rule algorithm first calculates the range that each influencer s's influence contribution falls in among all possible cascades producing $\mathcal{T}$ by calling Algorithm 3. Specifically, the ranges of the Shapley values are derived as follows. Let a_s, b_s be the lower bound and upper bound of each influencer s's influence contribution respectively. We record the influencers that can reach each node $v \in T(S)$ in $S(v)$ (Lines 2–4). For each influencer $s \in S$, in any possible cascade, its influence contribution is upper bounded by the number of non-influencer nodes it can reach in D (Lines 6–7). Meanwhile, for each node $v \in T(S)$, if v is only reachable from one influencer s in S, we increase the lower bound of s by 1, since s will always contribute to influencing v to produce the given campaign result (Lines 8–10). Finally, we return the lower bound a_s and upper bound b_s for each influencer $s \in S$ (Line 11).

Using the stopping rule algorithm, to estimate a single random variable X in the range of $[a, b]$ with $\mathbb{E}[X] = \mu$, we repeatedly generate samples until the sum of the observed values reaches the threshold Υ, where $\Upsilon = 2(b-a)(1+\varepsilon)(\frac{b-a}{b} + \frac{1}{3}\varepsilon)\ln(\frac{2}{\delta})\frac{1}{\varepsilon^2}$. The estimated value $\tilde{\mu} = \Upsilon/\theta$ (where θ is the number of samples generated) is an (ε, δ)-approximation of μ, i.e., it satisfies $\Pr[(1-\varepsilon)\mu \le \tilde{\mu} \le (1+\varepsilon)\mu] \ge 1 - \delta$ [47]. As we need to estimate $|S|$ expected influence contributions simultaneously in our problem, we calculate a threshold Υ_s for each expected influence contribution $\psi(s)$ for $s \in S$. As shown in Algorithm 4, Υ_s is calculated based on the accuracy guarantee (ε, δ) and the range of $[a_s, b_s]$ returned by

Algorithm 4: Stopping Rule Algorithm

```
   Input: influencer set $S$, cascade graph $D$, observation $\mathcal{T}$,
          and accuracy parameters $(\varepsilon, \delta)$
   Output: an $(\varepsilon, \delta)$-approximation $\tilde{\psi}(s)$ of the expected
           influence contribution for each influencer $s \in S$
1  obtain $\{(a_s, b_s) : s \in S\}$ via Algorithm 3;
2  foreach $s \in S$ do
3      $\Upsilon_s \leftarrow 2(b_s - a_s)(1 + \varepsilon)(\frac{b_s - a_s}{b_s} + \frac{1}{3}\varepsilon)\ln(\frac{2}{\delta})\frac{1}{\varepsilon^2}$;
4  initialize $\Sigma_s \leftarrow 0$ for each $s \in S$ and $\theta \leftarrow 0$;
5  while $\exists \Sigma_s < \Upsilon_s$ do
6      $\theta \leftarrow \theta + 1$;
7      obtain $\{\psi_{\omega_\theta}(s) : s \in S\}$ via Algorithm 2;
8      foreach influencer $s \in S$ do
9          if $\Sigma_s < \Upsilon_s$ then
10             $\Sigma_s \leftarrow \Sigma_s + \psi_{\omega_\theta}(s)$;
11             $\theta_s \leftarrow \theta$;
12 return $\{\tilde{\psi}(s) = \frac{\Upsilon_s}{\theta_s} : s \in S\}$;
```

Algorithm 3 for each influencer $s \in S$ (Line 3). Then, the cascade samples are iteratively generated by Algorithm 2 until the sum Σ_s exceeds Υ_s (Lines 5–11). We record the number of cascade samples θ_s when Σ_s exceeds Υ_s for each influencer $s \in S$ (Line 11). Finally, the estimation $\tilde{\psi}(s) = \Upsilon_s / \theta_s$ is returned for each influencer $s \in S$ (Line 12). The following lemma presents the accuracy guarantee of the estimations returned by Algorithm 4.

Lemma 4.3 ([47]). *Algorithm 4 returns an (ε, δ)-approximation $\tilde{\psi}(s)$ for each influencer $s \in S$, i.e.,*

$$\Pr\left[(1 - \varepsilon)\psi(s) \le \tilde{\psi}(s) \le (1 + \varepsilon)\psi(s)\right] \ge 1 - \delta.$$

To ensure all the values $\psi(s)$'s are estimated accurately, by a union bound, we have

$$\Pr\left[\bigwedge_{s \in S} (1 - \varepsilon)\psi(s) \le \tilde{\psi}(s) \le (1 + \varepsilon)\psi(s)\right] \ge 1 - |S|\delta.$$

Thus, to ensure the estimation accuracy with a high probability of $1 - \delta$, we can simply scale δ by a factor of $1/|S|$ as an input to Algorithm 4.

Time Complexity. Algorithm 3 takes $O(|S| \cdot m_D)$ time to perform a union operation for each edge. It can be inferred from Lemma 4.3 that $\theta \le \max_{s \in S} \frac{\Upsilon_s}{(1-\varepsilon)\psi(s)}$ with a high probability of $1 - |S|\delta$. This indicates that Algorithm 2 is invoked at most $\max_{s \in S} \frac{\Upsilon_s}{(1-\varepsilon)\psi(s)}$ times with a high probability, which requires $O(|S| \cdot \mathrm{TE} + m_D)$ time for each call in expectation. Meanwhile, updating Σ_s for all S takes $O(|S|)$ time. Therefore, with a high probability, Algorithm 4 takes $O\big((|S| \cdot \mathrm{TE} + m_D) \cdot \max_{s \in S} \frac{\Upsilon_s}{\psi(s)}\big)$ time in expectation. We assume that $\psi(s) > 0$ for each influencer $s \in S$, i.e., s would have at least one neighbor u (otherwise we can exclude s from S and analyze the influencer contributions among the remaining influencers). Then, $\psi(s)$ is at least $p_{s,u}/|S|$ in the worst case when $p_{s',u} = 1$ for each influencer $s' \in I_u$, i.e., $O\big(\psi(s)\big)$ is at least $O\big(\mathrm{poly}(\frac{1}{|V|})\big)$ when $p_{s,u}$ is a constant or $O\big(\mathrm{poly}(\frac{1}{|V|})\big)$ where $|V|$ is the number of nodes in graph G. As a result, Algorithm 4 is an FPRAS.

5 RELATED WORK

Influence estimation and its applications have been extensively studied in the literature [1–5, 7–12, 14, 17, 19, 20, 22–26, 28–33, 35–43, 48]. In viral marketing, influence maximization is a key algorithmic problem first studied by Domingos and Richardson [12, 33]. Then, Kempe et al. [25] formulated several diffusion models and a greedy algorithm was proposed to tackle the problem based on submodularity. After that, many follow-up works have been done on improving the efficiency and scalability of influence maximization on large-scale social networks [1, 5, 9–11, 19, 24, 30–32, 36, 37, 40, 41]. The reverse influence sampling approach [5] is widely used for estimating the influence spread in the domain of influence maximization. Some studies extended the vanilla influence maximization problem by taking the cost of activating the seed influencer into account [4, 29, 43], in which every influencer is associated with a fixed threshold value that indicates the amount of cost to activate the influencer to initiate the campaign. These costs are given a priori and are not necessarily relevant to the influence contributions. There is also a line of algorithmic viral marketing research focusing on allocating the discounts to the seed influencers assuming that whether an influencer can be activated as a seed influencer is uncertain [21, 44–46]. Under this setting, the probability that the influencer takes the bid of discount to initiate the campaign follows the purchase probability curve—the larger the discount, the higher the probability to purchase. The allocated discounts can hardly be adopted to capture the influence contributions in our problem in that the influencers who initiated the campaign are deterministic given the campaign result while the discounts are also assigned to the influencers who did not participate in the campaign. In incentivized social advertising, the incentives to initiate the campaigns are often derived according to the influence spreads of the influencers [2, 3]. These studies aimed to minimize the regret and maximize the revenue from the OSN provider's perspective by controlling the assignment of advertising campaigns. However, as discussed earlier, influence spread cannot precisely capture the contributions given the set of nodes activated in a particular campaign.

The Shapley value [34] in cooperative game theory is the solution concept that provides a fair way of dividing the value of the grand coalition. When k players participate in the game, the naive way to calculate the Shapley value for each player is to enumerate all $k!$ permutations of the players, which requires intensive computation and is generally intractable. There are some studies on developing bounded approximate solutions. Castro et al. [6] proposed a sampling-based algorithm for the case where the variance or the range of marginal contributions of a player is known. Liben-Nowell et al. [27] proposed a sampling-based algorithm for supermodular games that runs in polynomial time of the number of players. Chen et al. [8] proposed several influence-based centrality measures for stochastic graphs, including an influence-based Shapley centrality from the group perspective.

In this paper, we measure the Shapley values of the influencers characterizing their contributions in a given campaign result, which is fundamentally different from the existing literature focusing on the vanilla influence spreads of nodes. As our sample space consists of only the cascades in which a given set of nodes are activated, the existing sampling methods are no longer efficient or practical to generate relevant cascades and achieve accuracy guarantees.

Table 1: Datasets.

Dataset	#nodes	#edges	Avg. degree	Type
Facebook	4.0K	88.2K	43.7	Undirected
Google+	107.6K	13.7M	254.1	Directed
LiveJournal	4.8M	69.0M	28.5	Directed
Orkut	3.1M	117.2M	76.3	Undirected

6 EXPERIMENTS

This section experimentally evaluates the quality and efficiency of our proposed algorithms. We implement our algorithms using C++. All experiments are run on a machine with Intel Xeon 2.4GHz CPU and 384GB memory.

6.1 Experimental Setup

Datasets. Several real datasets including Facebook, Google+, LiveJournal and Orkut are used to evaluate our proposed algorithms. All the datasets are available at http://snap.stanford.edu/data. Table 1 gives the details of these datasets.

Compared Algorithms. We compare the influence contributions delivered by Algorithm 1 and Algorithm 4 with an accuracy of $(0.5, 0.1)$-approximation against the following baselines.

- Uniform: The contributions are allocated uniformly among the influencers in S.
- Degree: The contributions are allocated according to the degree distribution of influencers with DEG-1 being the degree distribution in the original social network graph G and DEG-2 being the degree distribution in the extracted subgraph D.
- Influence Spreads: The contributions are allocated according to the influence spread distribution of influencers with INF-1 being the influence spread in the original social network graph G and INF-2 being the influence spread in the extracted subgraph D. We adopt the stopping rule algorithm to obtain a $(0.1, 0.05)$-approximation of the influence spreads.

The total contributions of the above methods are all normalized to $|T(S)|$, since a total number $|T(S)|$ of nodes are influenced.

Ground Truth. The propagation probability $p_{u,v}$ of each edge (u, v) is set to the reciprocal of v's in-degree which is a commonly used setting by other studies [36, 41]. The influencer sets of different sizes are experimented. Given an influencer set size k, we select the influencer set consisting of the top k nodes with the highest out-degrees. We perform Monte Carlo simulations to generate the ground truth. We generate both discrete and continuous activation times for the IC model and only continuous activation times for the LT model (since the result is the same for discrete times under the LT model as discussed in Section 3.5). For the discrete setting, we perform the standard Monte Carlo simulations according to the IC diffusion model to record the true cascades. Then, we calculate the exact influence contribution for each influencer according to Definition 2.1. For the continuous setting under the IC model, for each outgoing edge (u, v) starting from the influencer set S, we initialize a random activation trial time referring to the activation trial time of v and maintain a minimum heap h_t of the trial times. The trial time is randomly chosen from an exponential distribution with a probability density function of $f(x) = e^{-x}$ [16]. The edge (u, v) with the minimum activation time is popped out from h_t. If v is not activated, (u, v) is live with a probability of $p_{u,v}$. Otherwise, we do not consider the edge and continue the simulation process. If (u, v) is live, we further generate random trial times for v's inactive neighbors (i.e., v's activation time adding a random number from the exponential distribution) and push their trial times into h_t. The above process is repeated until no further nodes can be activated, i.e., h_t is empty. We get the number of nodes in $T(S)$ that each influencer activates (directly or indirectly) in the advertising campaign as the ground truth. Under the LT model, we slightly modify the simulation process as follows. At the beginning, we randomly select a threshold $\lambda_v \in [0, 1]$ for each node $v \in V$. When (u, v) is popped out from h_t, we check whether the total weight of v's activated inverse neighbors reach λ_v (i.e., whether v is activated) and other procedures remain the same. Finally, the influence contribution returned by Algorithm 1 is the ground truth.

Performance Metric. For each ground truth of cascade, we measure the mean-squared-error (MSE) of influence contributions, i.e.,

$$\frac{1}{|S|} \sum_{s \in S} \left(\tilde{\psi}(s) - \psi_\omega(s) \right)^2, \tag{14}$$

where $\tilde{\psi}(s)$ is the influence contribution of s obtained by any algorithm, and $\psi_\omega(s)$ is the ground truth. We repeat the experiments for 10,000 times for Facebook and Google+ and 100 times for LiveJournal and Orkut, and report the average results.

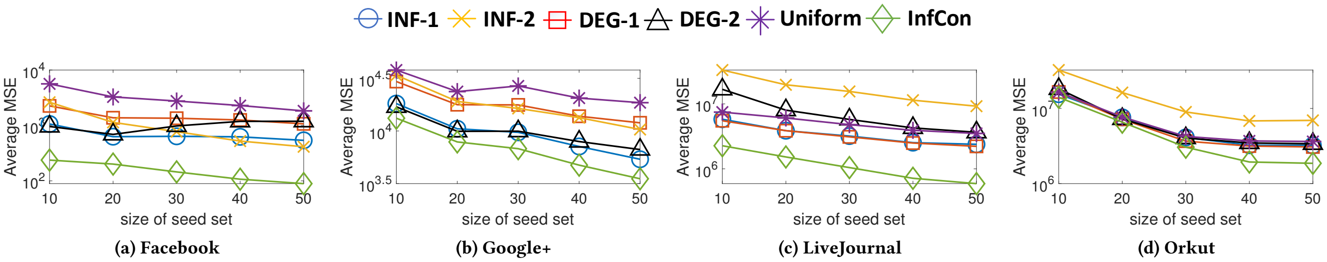
6.2 Experimental Result

6.2.1 Influence Contribution. Figure 5 shows the result under the IC model when the conventional discrete timestamp is considered, where our method is denoted by InfCon. The average MSE of our method is significantly smaller than all the baselines for all the datasets tested. In particular, for the Facebook dataset, our method produces 2 orders of magnitudes smaller average MSE than the Uniform method. Such a result demonstrates the superiority of our proposed solution.

Figure 6 gives the result under the IC model when continuous activation time is considered. The result is almost the same as that under the discrete setting. This verifies that our solution is also a great estimator that can characterize the influence contributions rather well compared to other baselines in practical scenarios.

Figure 7 shows the result under the LT model. Note that we can compute the exact influence contributions under the LT model, i.e., the ground truth, so our method has an average MSE of 0. We observe that the INF-2 method tends to have a small average MSE under the LT model for various datasets while it produces a large average MSE under the IC model. This is because the INF-2 method can hardly tackle influence overlaps under the IC model and it can characterize the influence contributions well without influence overlaps under the LT model.

6.2.2 Sampling Efficiency. We run Algorithm 4 with a more accurate approximation guarantee of $(0.1, 0.05)$ to show the sampling



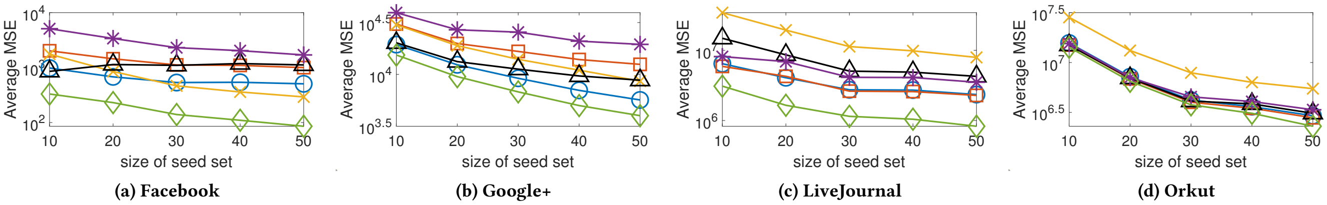
(a) Facebook

(b) Google+

(c) LiveJournal

(d) Orkut

Figure 5: Average mean-squared-error under the IC model (discrete timestamp).



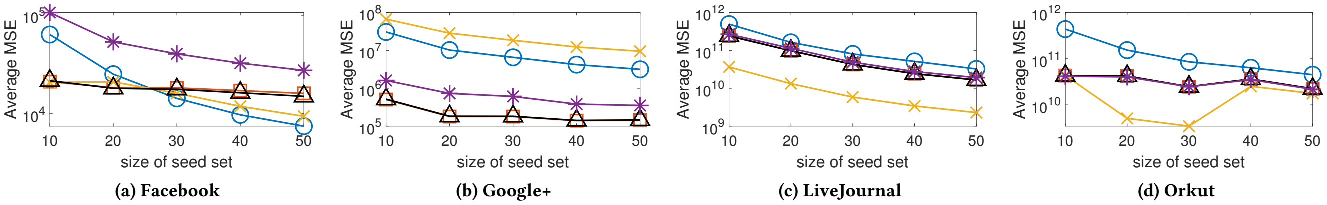
(a) Facebook

(b) Google+

(c) LiveJournal

(d) Orkut

Figure 6: Average mean-squared-error under the IC model (continuous time).



(a) Facebook

(b) Google+

(c) LiveJournal

(d) Orkut

Figure 7: Average mean-squared-error under the LT model (continuous time).

Table 2: Scaling factor $\frac{1}{\prod_{v\in T(S)}\beta_v}$ on top-degree influencers.

#influencers	Facebook	Google+	LiveJournal	Orkut
10	$10^{602.2}$	$10^{3828.7}$	$10^{37405.9}$	$10^{100439.5}$
20	$10^{630.0}$	$10^{5198.0}$	$10^{48094.4}$	$10^{126132.0}$
30	$10^{635.9}$	$10^{6106.5}$	$10^{53908.3}$	$10^{145614.0}$
40	$10^{666.4}$	$10^{6675.1}$	$10^{65343.6}$	$10^{171359.2}$
50	$10^{678.8}$	$10^{7007.1}$	$10^{71381.0}$	$10^{188532.0}$

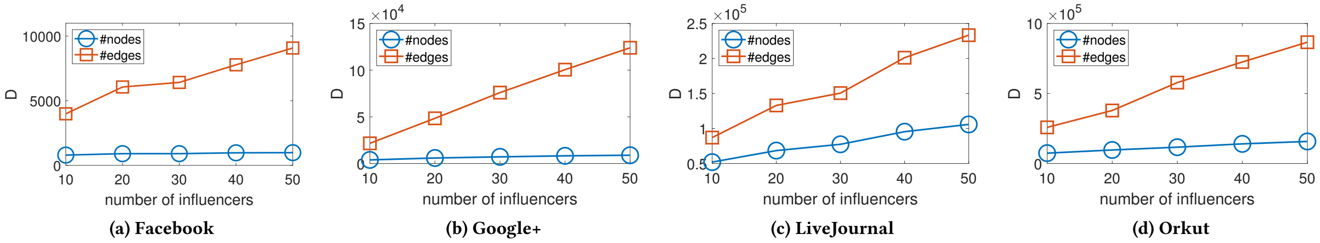
Table 3: Ratio of $\frac{\Upsilon_0}{\Upsilon}$ on top-degree influencers.

#influencers	Facebook	Google+	LiveJournal	Orkut
10	7.38	1.43	1.95	1.20
20	5.48	1.43	1.95	1.24
30	4.13	1.45	2.16	1.25
40	4.98	1.46	2.18	1.26
50	5.07	1.46	2.27	1.27

efficiency under the IC model when continuous activation time is considered. Tables 2–3 and Figures 8–9 show the average result of 10 repeated experiment runs with a more accurate approximation setting of $(0.1, 0.05)$ for different datasets.

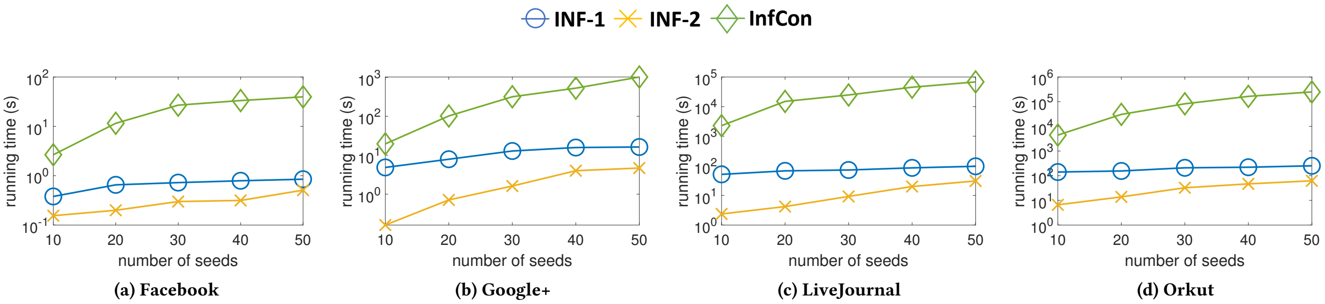
Scaling Factor. Table 2 shows the scaling factor $\frac{1}{\prod_{v\in T(S)}\beta_v}$ of sample space given in Proposition 3.5. As can be seen, the scaling factor is very high, indicating that our sample space is many orders of magnitude smaller than the original sample space in the cascade graph D. The naive reverse influence sampling method or Monte Carlo method cannot finish to produce the results with the same accuracy guarantee even when the campaign result is relatively small consisting of hundreds of nodes only.

Ratio of $\frac{\Upsilon_0}{\Upsilon}$. We calculate the threshold Υ_s for each influencer s in Algorithm 4 based on the bounds of $[a_s, b_s]$ and compare the threshold $\Upsilon = \max_{s\in S}\Upsilon_s$ with the vanilla threshold Υ_0 with lower bound $a = 0$ and upper bound $b = |T(S)|$. We show the ratio of $\frac{\Upsilon_0}{\Upsilon}$ in Table 3. It can be seen that Υ_0 is a few times larger than Υ. As the number of cascade samples required to yield an estimation with the given accuracy guarantee is linear to the threshold value, it indicates that our derived bounds of $[a_s, b_s]$ for each influencer $s \in S$ can effectively boost the sampling efficiency with less cascade samples needed. Note that the trend of Υ_0/Υ under Facebook is different from other datasets. This is because the size of the Facebook dataset is relatively small compared with other datasets and a few high-degree nodes can activate a large portion of all nodes, i.e., b does not change much and hence Υ_0 do not increase notably with increasing number of influencers. Meanwhile, a_s decreases when more influencers are selected to start the campaign due to more overlapping reachable nodes and thus Υ becomes larger. The combination of these effects result in a different trend of $\frac{\Upsilon_0}{\Upsilon}$ for the Facebook dataset.



(a) Facebook (b) Google+ (c) LiveJournal (d) Orkut

Figure 8: Cascade graph D on top-degree influencers.



(a) Facebook (b) Google+ (c) LiveJournal (d) Orkut

Figure 9: Sampling time (seconds) on top-degree influencers.

Cascade Graph. We can see from Figure 8 that the number of edges in the cascade graph D extracted is much larger than that of nodes and each node has a considerable number of neighbors on average, which indicates that the sample space, consisting of all possible cascades producing the observed result $\mathcal{T}$, is huge.

Running Time. As shown in Table 2, if we directly generate samples by the naive reverse influence sampling method or Monte Carlo method, it is (almost) impossible to reproduce the observed campaign result, i.e., less than a probability of $1/10^{600}$ on Facebook, $1/10^{3000}$ on Google+, $1/10^{30000}$ on LiveJournal, and $1/10^{100000}$ on Orkut. This indicates that the naive reverse influence sampling method or Monte Carlo method cannot finish to produce the estimation results with the same accuracy guarantee. To evaluate the efficiency of our InfCon algorithm, Figure 9 shows the running time of InfCon against two heuristic algorithms including INF-1 and INF-2. (Note that we do not show the running time of DEG-1, DEG-2 and Uniform, as these heuristics take near zero time.) We observe that the INF-1 and INF-2 generally run faster than our InfCon algorithm, since INF-1 and INF-2 use the influence spread of each node to roughly represent the influence contribution and do not consider the actual campaign result. However, as demonstrated in Figures 5–7, both INF-1 and INF-2 perform poorly in characterizing the influence contributions. In fact, by sampling cascades consistent with $\mathcal{T}$ following the probability distribution of $\Pr[\omega \sim \mathcal{T}]$, our algorithm efficiently estimates the influence contributions in a reasonable amount of time for all the cases tested. Meanwhile, as can be also seen from Figure 9, the running time of our algorithm generally increases with the influencer set size. This is because the campaign result, i.e., the set of nodes activated, becomes larger when there are more influencers and thus the cascade graph extracted includes more nodes and edges (Figure 8).

7 CONCLUSION

In this paper, we propose a new metric, i.e., influence contribution, to measure the influencers' contributions given the result of an advertising campaign, based on which we formulate a problem of influence contribution allocation (ICA). We show that the Shapley value provides the exact solution for the ICA problem. Moreover, to address ICA effectively and efficiently, a linear time algorithm is developed to find the exact solution under the LT model and an FPRAS is devised to construct an approximate solution under the IC model. Our solution under the IC model consists of a scalable sampling method that significantly boosts the sampling efficiency and a stopping rule algorithm that delivers an approximate solution with accuracy guarantees. Through extensive experiments, we show significant efficiency and efficacy improvements of our approach against other baselines.

ACKNOWLEDGMENTS

Jing Tang's work is partially supported by HKUST(GZ) under a Startup Grant. Xueyan Tang's work is partially supported by Singapore Ministry of Education Academic Research Fund Tier 1 under Grant 2019-T1-002-042. Lei Chen's work is partially supported by National Key Research and Development Program of China Grant No. 2018AAA0101100, the Hong Kong RGC GRF Project 16207617, CRF Project C6030-18G, C1031-18G, C5026-18G, RIF Project R6020-19, AOE Project AoE/E-603/18, Theme-based project TRS T41-603/20R, China NSFC No. 61729201, Guangdong Basic and Applied Basic Research Foundation 2019B151530001, Hong Kong ITC ITF grants ITS/044/18FX and ITS/470/18FX, Microsoft Research Asia Collaborative Research Grant, HKUST-NAVER/LINE AI Lab, HKUST-Webank joint research lab grants.

REFERENCES

[1] Akhil Arora, Sainyam Galhotra, and Sayan Ranu. 2017. Debunking the Myths of Influence Maximization: An In-Depth Benchmarking Study. In *Proc. ACM SIGMOD.* 651–666.

[2] Cigdem Aslay, Francesco Bonchi, Laks V.S. Lakshmanan, and Wei Lu. 2017. Revenue Maximization in Incentivized Social Advertising. *Proc. VLDB Endowment* 10, 11 (2017), 1238–1249.

[3] Cigdem Aslay, Wei Lu, Francesco Bonchi, Amit Goyal, and Laks V.S. Lakshmanan. 2015. Viral Marketing Meets Social Advertising: Ad Allocation with Minimum Regret. *Proc. VLDB Endowment* 8, 7 (2015), 814–825.

[4] Song Bian, Qintian Guo, Sibo Wang, and Jeffrey Xu Yu. 2020. Efficient Algorithms for Budgeted Influence Maximization on Massive Social Networks. *Proc. VLDB Endowment* 13, 9 (2020), 1498–1510.

[5] Christian Borgs, Michael Brautbar, Jennifer Chayes, and Brendan Lucier. 2014. Maximizing Social Influence in Nearly Optimal Time. In *Proc. SODA.* 946–957.

[6] Javier Castro, Daniel Gómez, and Juan Tejada. 2009. Polynomial Calculation of the Shapley Value Based on Sampling. *Computers & Operations Research* 36, 5 (2009), 1726–1730.

[7] Shuo Chen, Ju Fan, Guoliang Li, Jianhua Feng, Kian-Lee Tan, and Jinhui Tang. 2015. Online Topic-Aware Influence Maximization. *Proc. VLDB Endowment* 8, 6 (2015), 666–677.

[8] Wei Chen and Shang-Hua Teng. 2017. Interplay Between Social Influence and Network Centrality: A Comparative Study on Shapley Centrality and Single-Node-Influence Centrality. In *Proc. WWW.* 967–976.

[9] Wei Chen, Chi Wang, and Yajun Wang. 2010. Scalable Influence Maximization for Prevalent Viral Marketing in Large-Scale Social Networks. In *Proc. ACM KDD.* 1029–1038.

[10] Wei Chen, Yajun Wang, and Siyu Yang. 2009. Efficient Influence Maximization in Social Networks. In *Proc. ACM KDD.* 199–208.

[11] Wei Chen, Yifei Yuan, and Li Zhang. 2010. Scalable Influence Maximization in Social Networks Under the Linear Threshold Model. In *Proc. IEEE ICDM.* 88–97.

[12] Pedro Domingos and Matt Richardson. 2001. Mining the Network Value of Customers. In *Proc. ACM KDD.* 57–66.

[13] Xing Fang. 2017. Understanding deep learning via backtracking and deconvolution. *Journal of Big Data* 4, 1 (2017), 1–14.

[14] Sainyam Galhotra, Akhil Arora, and Shourya Roy. 2016. Holistic Influence Maximization: Combining Scalability and Efficiency with Opinion-Aware Models. In *Proc. ACM SIGMOD.* 743–758.

[15] Michael R. Garey and David S. Johnson. 1990. *Computers and Intractability; A Guide to the Theory of NP-Completeness.* W. H. Freeman & Co., USA.

[16] Manuel Gomez-Rodriguez, Jure Leskovec, and Andreas Krause. 2012. Inferring Networks of Diffusion and Influence. *ACM Transactions on Knowledge Discovery from Data* 5, 4 (2012), 1–37.

[17] Amit Goyal, Francesco Bonchi, and Laks V.S. Lakshmanan. 2011. A Data-Based Approach to Social Influence Maximization. *Proc. VLDB Endowment* 5, 1 (2011), 73–84.

[18] Anirudh Goyal, Philemon Brakel, William Fedus, Soumye Singhal, Timothy P. Lillicrap, Sergey Levine, Hugo Larochelle, and Yoshua Bengio. 2019. Recall Traces: Backtracking Models for Efficient Reinforcement Learning. In *Proc. ICLR.*

[19] Qintian Guo, Sibo Wang, Zhewei Wei, and Ming Chen. 2020. Influence Maximization Revisited: Efficient Reverse Reachable Set Generation with Bound Tightened. In *Proc. ACM SIGMOD.* 2167–2181.

[20] Kai Han, Keke Huang, Xiaokui Xiao, Jing Tang, Aixin Sun, and Xueyan Tang. 2018. Efficient Algorithms for Adaptive Influence Maximization. *Proc. VLDB Endowment* 11, 9 (2018), 1029–1040.

[21] Kai Han, Chaoting Xu, Fei Gui, Shaojie Tang, He Huang, and Jun Luo. 2018. Discount Allocation for Revenue Maximization in Online Social Networks. In *Proc. ACM Mobihoc.* 121–130.

[22] Keke Huang, Jing Tang, Kai Han, Xiaokui Xiao, Wei Chen, Aixin Sun, Xueyan Tang, and Andrew Lim. 2020. Efficient Approximation Algorithms for Adaptive Influence Maximization. *The VLDB Journal* 29, 6 (2020), 1385–1406.

[23] Keke Huang, Jing Tang, Xiaokui Xiao, Aixin Sun, and Andrew Lim. 2020. Efficient Approximation Algorithms for Adaptive Target Profit Maximization. In *Proc. IEEE ICDE.* 649–660.

[24] Keke Huang, Sibo Wang, Glenn Bevilacqua, Xiaokui Xiao, and Laks V.S. Lakshmanan. 2017. Revisiting the Stop-and-Stare Algorithms for Influence Maximization. *Proc. VLDB Endowment* 10, 9 (2017), 913–924.

[25] David Kempe, Jon Kleinberg, and Éva Tardos. 2003. Maximizing the Spread of Influence Through a Social Network. In *Proc. ACM KDD.* 137–146.

[26] Yuchen Li, Dongxiang Zhang, and Kian-Lee Tan. 2015. Real-time Targeted Influence Maximization for Online Advertisements. *Proc. VLDB Endowment* 8, 10 (2015), 1070–1081.

[27] David Liben-Nowell, Alexa Sharp, Tom Wexler, and Kevin Woods. 2012. Computing Shapley Value in Supermodular Coalitional Games. In *Proc. COCOON.* 568–579.

[28] Wei Lu, Wei Chen, and Laks V.S. Lakshmanan. 2015. From Competition to Complementarity: Comparative Influence Diffusion and Maximization. *Proc. VLDB Endowment* 9, 2 (2015), 60–71.

[29] Huy Nguyen and Rong Zheng. 2013. On Budgeted Influence Maximization in Social Networks. *IEEE Journal on Selected Areas in Communications* 31, 6 (2013), 1084–1094.

[30] Hung T. Nguyen, My T. Thai, and Thang N. Dinh. 2016. Stop-and-Stare: Optimal Sampling Algorithms for Viral Marketing in Billion-Scale Networks. In *Proc. ACM SIGMOD.* 695–710.

[31] Naoto Ohsaka. 2020. The Solution Distribution of Influence Maximization: A High-Level Experimental Study on Three Algorithmic Approaches. In *Proc. ACM SIGMOD.* 2151–2166.

[32] Naoto Ohsaka, Tomohiro Sonobe, Sumio Fujita, and Ken-ichi Kawarabayashi. 2017. Coarsening Massive Influence Networks for Scalable Diffusion Analysis. In *Proc. ACM SIGMOD.* 635–650.

[33] Matthew Richardson and Pedro Domingos. 2002. Mining Knowledge-Sharing Sites for Viral Marketing. In *Proc. ACM KDD.* 61–70.

[34] Lloyd S Shapley. 1953. A Value for n-Person Games. *Contributions to the Theory of Games* 2, 28 (1953), 307–317.

[35] Jing Tang, Keke Huang, Xiaokui Xiao, Laks V.S. Lakshmanan, Xueyan Tang, Aixin Sun, and Andrew Lim. 2019. Efficient Approximation Algorithms for Adaptive Seed Minimization. In *Proc. ACM SIGMOD.* 1096–1113.

[36] Jing Tang, Xueyan Tang, Xiaokui Xiao, and Junsong Yuan. 2018. Online Processing Algorithms for Influence Maximization. In *Proc. ACM SIGMOD.* 991–1005.

[37] Jing Tang, Xueyan Tang, and Junsong Yuan. 2018. An Efficient and Effective Hop-Based Approach for Inluence Maximization in Social Networks. *Social Network Analysis and Mining* 8, 10 (2018).

[38] Jing Tang, Xueyan Tang, and Junsong Yuan. 2018. Profit Maximization for Viral Marketing in Online Social Networks: Algorithms and Analysis. *IEEE Transactions on Knowledge and Data Engineering* 30, 6 (2018), 1095–1108.

[39] Jing Tang, Xueyan Tang, and Junsong Yuan. 2018. Towards Profit Maximization for Online Social Network Providers. In *Proc. IEEE INFOCOM.* 1178–1186.

[40] Youze Tang, Yanchen Shi, and Xiaokui Xiao. 2015. Influence Maximization in Near-Linear Time: A Martingale Approach. In *Proc. ACM SIGMOD.* 1539–1554.

[41] Youze Tang, Xiaokui Xiao, and Yanchen Shi. 2014. Influence Maximization: Near-Optimal Time Complexity Meets Practical Efficiency. In *Proc. ACM SIGMOD.* 75–86.

[42] Yanhao Wang, Qi Fan, Yuchen Li, and Kian-Lee Tan. 2017. Real-time Influence Maximization on Dynamic Social Streams. *Proc. VLDB Endowment* 10, 7 (2017), 805–816.

[43] Yue Wang, WeiJing Huang, Lang Zong, TengJiao Wang, and DongQing Yang. 2013. Influence maximization with limit cost in social network. *Science China Information Sciences* 56, 7 (2013), 1–14.

[44] Yu Yang, Xiangbo Mao, Jian Pei, and Xiaofei He. 2016. Continuous Influence Maximization: What Discounts Should We Offer to Social Network Users?. In *Proc. ACM SIGMOD.* 727–741.

[45] Jing Yuan and Shao-Jie Tang. 2017. Adaptive Discount Allocation in Social Networks. In *Proc. ACM Mobihoc.* Article 22.

[46] Bo-Lei Zhang, Zhu-Zhong Qian, Wen-Zhong Li, Bin Tang, Sang-Lu Lu, and Xiaoming Fu. 2016. Budget allocation for maximizing viral advertising in social networks. *Journal of Computer Science and Technology* 31, 4 (2016), 759–775.

[47] Yuqing Zhu, Jing Tang, and Xueyan Tang. 2020. Pricing Influential Nodes in Online Social Networks. *Proc. VLDB Endowment* 13, 10 (2020), 1614–1627.

[48] Yuqing Zhu, Jing Tang, Xueyan Tang, Sibo Wang, and Andrew Lim. 2021. 2-hop+ Sampling: Efficient and Effective Influence Estimation. *IEEE Transactions on Knowledge and Data Engineering* (2021).