Scalable Lattice Influence Maximization

Wei Chen, *Fellow, IEEE*, Ruihan Wu, and Zheng Yu

***Abstract*— Influence maximization is the task of finding k seed nodes in a social network such that the expected number of activated nodes in the network (under certain influence propagation model), referred to as the influence spread, is maximized. Lattice influence maximization (LIM) generalizes influence maximization such that, instead of selecting k seed nodes, one selects a vector $\mathbf{x} = (x_1, \ldots, x_d)$ from a discrete space $\mathcal{X}$ called a lattice, where x_j corresponds to the jth marketing strategy and $\mathbf{x}$ represents a marketing strategy mix. Each strategy mix $\mathbf{x}$ has probability $h_u(\mathbf{x})$ to activate a node u as a seed. LIM is the task of finding a strategy mix under the constraint $\sum_j x_j \le k$ such that its influence spread is maximized. We adopt the reverse influence sampling (RIS) approach and design scalable algorithms for LIM. We explore two complementary design choices: one algorithm IMM-PRR is based on partial coverage on reverse-reachable sets, and the other IMM-VSN is based on incorporating virtual strategy nodes. IMM-PRR can be applied as a general solution to LIM, and we further improve its efficiency for a large family of models where each strategy independently activates seed nodes. IMM-VSN is explicitly designed for the case of independent strategy activation, and it uses virtual nodes to represent strategies to reduce LIM back to the original influence maximization problem. We prove that both IMM-PRR and IMM-VSN guarantee $1 - 1/e - \varepsilon$ approximation for small $\varepsilon > 0$. We further extend LIM to the partitioned budget case where strategies are partitioned into groups, each of which has a separate budget, and show that a minor variation of our algorithms would achieve $1/2 - \varepsilon$ approximation ratio with the same time complexity. Empirically, through extensive tests, we demonstrate that IMM-VSN runs faster than IMM-PRR and much faster than other baseline algorithms while providing the same level of influence spread. We conclude that IMM-VSN is the best one for models with independent strategy activations, while IMM-PRR works for general modes without this assumption.**

***Index Terms*— Influence maximization, lattice influence maximization (LIM), reverse influence sampling (RIS), scalable influence maximization.**

I. Introduction

THE classical influence maximization task is to find a small set of seed nodes to maximize the expected number of activated nodes from these seeds, referred to as the influence spread, based on the certain diffusion process in a social network [1]. It models the viral marketing scenario in social networks and its variants also find applications in diffusion monitoring, rumor control, crime prevention, and so on [2]–[5]. Therefore, numerous studies on influence maximization have been conducted since its inception. One important direction is scalable influence maximization, which aims to design efficient approximation algorithms and heuristics for large social networks. Many diverse approaches, including graph theoretic heuristics, sketching methods, and random sampling, have been tried for scalable influence maximization [6]–[12]. Other directions include competitive and complementary influence maximization [3], [4], [13], continuous-time influence maximization [14], topic-aware influence maximization [15], and so on.

However, a generalization of influence maximization already considered by Kempe *et al.* [1] in their seminal paper receives much less attention and is left largely unexplored. Kempe *et al.* considered viral marketing scenarios with a general marketing strategy mix of d different strategies, with each strategy j taking value x_j (e.g., money put into strategy j). The combined strategy mix is a vector $\mathbf{x} = (x_1, x_2, \ldots, x_d)$. When applying the strategy mix $\mathbf{x}$ to the social network, each node u in the social network has a probability of $h_u(\mathbf{x})$ to be activated as a seed. After the seeds are probabilistically activated by the marketing strategies, influence propagates from the seeds in the network as dictated by an influence diffusion model. The optimization problem is to find the best strategy mix $\mathbf{x}^*$ that maximizes the influence spread subject to the budget constraint $\sum_{j\in[d]} x_j \le k$, where notation $[d]$ means $\{1, 2, \ldots, d\}$. In this article, we consider strategy mixes taken from a discrete space $\mathcal{X}$ referred to as a lattice, and thus, we call the abovementioned optimization problem, lattice influence maximization (LIM).

LIM represents more realistic scenarios since, in practice, companies often apply a mix of marketing strategies, such as coupons, direct mails, marketing events, and target different segments of users. Kempe *et al.* [1] outlined the basic approach based on submodularity and greedy algorithm to solve the problem. This direction, however, has not been further investigated in the research community. The only relevant study that we find is [16] that investigates influence maximization with fractional or continuous discounts on users in the network, a special case of the LIM problem.

In this article, we provide a detailed study of the scalable solutions for the LIM problem. It is well known that the naive greedy approach for influence maximization is not scalable due to excessive Monte Carlo simulations. The problem could be even worse for LIM when we have a large strategy space with complicated interactions with the social network. We tackle this problem by adopting the reverse influence sampling (RIS) approach [9]–[11], which is successful for the classical influence maximization problem. The adaption of RIS to LIM is not straightforward because nodes in the network are not deterministically selected as seeds but probabilistically

Manuscript received November 3, 2019; revised April 3, 2020; accepted May 16, 2020. *(Corresponding author: Zheng Yu.)*

Wei Chen is with Microsoft Research Asia, Beijing 100080, China (e-mail: weic@microsoft.com).

Ruihan Wu is with the Department of Computer Science, Cornell University, Ithaca, NY 14853 USA (e-mail: rw565@cornell.edu).

Zheng Yu is with the Department of Operations Research and Financial Engineering, Princeton University, Princeton, NJ 08544 USA (e-mail: zhengy@princeton.edu).

Digital Object Identifier 10.1109/TCSS.2020.2998777

selected based on the complicated function $h_u(\mathbf{x})$. In fact, the study in [16] does not apply the RIS approach and only provides some heuristic algorithms without any theoretical guarantee.

In this article, we explore two complementary design choices and design two algorithms based on the RIS approach. The first approach extends the key idea in the RIS approach, which shows that the influence spread of seed set S equals to n multiplied by the probability of S covering a randomly generated reserve-reachable set (RR set) R, where n is the total number of nodes in the network and S covering R means S intersects with R. Our extension shows that the influence spread of a strategy mix $\mathbf{x}$ equals to n multiplied by a partial coverage of a random RR set R. From this extension, we design a general scalable algorithm IMM-PRR adapted from the influence maximization with martingales (IMM) algorithm for the classical influence maximization problem [11].

To improve the efficiency of IMM-PRR, we identify a large class of LIM problems in which each strategy could independently activate nodes as seeds in the social network, which we call independent strategy activation. We show that this class of problem covers many practical application scenarios, including user segment marketing, personalized marketing, and repeated event marketing.

The second design choice is to reduce the LIM problem into a classical influence maximization problem with seed set selection. To do so, we need to covert strategies into virtual strategy nodes (VSNs) and properly set up the propagation from these VSNs to real nodes to establish the equivalence. The reduction needs a novel integration of the classical independent cascade (IC) and linear threshold (LT) models into a single model to make it work, and this integration could be of independent interest by itself. We refer to the resulting algorithm as IMM-VSN.

For both IMM-PRR and IMM-VSN, we prove that they provide $1-1/e-\varepsilon$ approximation to the LIM problem for any $\varepsilon > 0$, and we analyze their time complexity, which indicates that IMM-VSN could perform better in running time.

We further generalize LIM originally proposed in [1] to accommodate partitioned budgets (denoted as the LIM-PB problem), that is, the strategies are partitioned into groups and each group has a separate budget. This matches the practical scenario when marketing activities are coordinated by multiple parties, each of which focusing on different marketing channels with different marketing budgets. We connect the LIM-PB problem with submodular maximization under matroid constraints, and thus, it implies that a minor variation of our algorithms would achieve $1/2-\varepsilon$ approximation ratio with the same time complexity.

We conduct extensive experiments of our algorithms and several baseline algorithms (including algorithms proposed in [16]) on five real-world networks with two different types of marketing strategies. Our experimental results demonstrate that IMM-VSN is faster than IMM-PRR and is much faster than all other baseline algorithms, while IMM-VSN and IMM-PRR provide the same or slightly better influence spread than other algorithms. Moreover, for both IMM-VSN/IMM-PRR, we can easily tune one parameter to balance between theoretical guarantee and faster performance.

In summary, we make the following contributions: 1) we explore two design approaches for scalable LIM solutions and propose two algorithms with theoretical guarantees: one is more general, and the other is more efficient in the case of independent strategy activations; 2) we extend the problem of LIM to the case of partitioned budgets and show that our scalable algorithms can still provide constant approximation; and 3) we demonstrate through experiments that IMM-VSN is the best for the case of independent strategy activations and runs much faster than other algorithms.

A. Related Work

Influence maximization for viral marketing is first studied as a data mining task in [17] and [18]. Kempe *et al.* [1] are the first to formulate the problem as a discrete optimization problem. They propose the IC, LT, triggering, and other more general models, study their submodularity, and propose the greedy algorithm that gives $1-1/e-\varepsilon$ approximate solution for $\varepsilon > 0$. They also propose an LIM problem and the greedy approach to solve the problem.

Scalable influence maximization is an important direction and receives much attention. Some early proposals rely on the properties of the IC and LT models as well as efficient graph algorithms to design scalable heuristics [7], [8], [19], [20]. Borgs *et al.* [9] proposed the novel approach of RIS that is able to provide both theoretical guarantee and scalable performance in practice. The RIS approach is improved by a series of studies [10], [11], [21], [22], which is also a demonstration that even with the known RIS approach achieving scalable influence maximization still requires significant design effort. Our algorithm is based on RIS and is adapted from the IMM algorithm [11]. The adaptations from other algorithms (e.g., DSSA [21] or OPIM [22]) would be similar, and we choose IMM mainly for its relative simplicity for illustrative purpose.

Many other directions of influence maximization have been studied, such as competitive and complementary influence maximization, and seed minimization. They are less relevant to our study, so we refer to a monograph [23] for more comprehensive coverage on influence maximization.

In terms of the LIM problem, the most relevant study is the one in [16]. In their model, each user could receive a personalized discount, which is translated to the probability of the user being activated as a seed. This corresponds to the personalized marketing scenario in our setting. They propose a scalable heuristic algorithm based on coordinate descent to solve the problem. Comparing with their study, our algorithm is better in: 1) providing theoretical guarantees on approximation ratio and running time; 2) solving a larger class of problems covering segment marketing, event marketing, and so on; and 3) outperforming their algorithm in both running time and influence spread.

Demaine *et al.* [24] proposed a fractional influence model, in which the fractional solution x_v for a node v affects not only on v's activation as a seed but also on v's activation by its neighbors during the diffusion process. Thus, their model

is incomparable with our LIM model although both allow fractional solutions.

DR-submodular function maximization over lattices or continuous domain receives much attention in recent years [25]–[27]. The continuous greedy algorithm of [25] provides a theoretical solution framework for solving continuous submodular maximization with various constraints and better approximation ratios, such as matroid or knapsack constraints, but the continuous greedy algorithm is too slow to be implemented in practice—it requires $O(n^5)$ iterations in its main greedy loop. Soma *et al.* [26] provided solutions for submodular maximization on a lattice with different submodular properties and constraints. The general L-Greedy algorithm (i.e., Algorithm 1) corresponds to their greedy algorithm for DR-submodular functions with uniform costs, and this algorithm is used as a baseline algorithm MCLG with Monte Carlo simulations for function evaluation. As shown by our experiments, this general version of the greedy algorithm is not scalable for our LIM problem. Their paper does talk about a scalable solution, but that is for a coverage problem in bipartite graphs, which does not apply to our LIM setting. Hassani *et al.* [27] applied the gradient method to continuous submodular maximization. The gradient method needs to assume that the objective function is continuous and differentiable, but our greedy solution does not require such assumptions. Moreover, it is nontrivial to adapt the gradient method to the context of influence maximization. Besides the abovementioned differences, all these studies focus on general DR-submodular functions, while our algorithmic design focuses on the specific DR-submodular function related to the influence maximization task.

Very recently, Chen *et al.* [28] studied the application of the gradient method to the problem of continuous influence maximization with budget saving (CIM-BS), which has the objective function as $g(\mathbf{x}) + \lambda(k - c(\mathbf{x}))$, where $g(\mathbf{x})$ is the influence spread of strategy mix $\mathbf{x}$, $c(\mathbf{x})$ is the cost of $\mathbf{x}$, k is the budget, and λ is a balance parameter. When $\lambda = 0$, the CIM-BS problem corresponds to our LIM problem except that our solution space is the discrete lattice. Their main result is that for this general objective function with $\lambda > 0$, their gradient-based algorithms could provide a constant approximation, but the greedy algorithm no longer has such a guarantee because the objective function is no long submodular. However, as they show, applying the gradient method is nontrivial, and their algorithms are not scalable to moderate or large networks.

II. Model and Problem Definition

Influence propagation in social networks is modeled by the triggering model [1]. A social network is modeled as a directed graph $G = (V, E)$, where V is the set of nodes representing individuals, and E is the set of directed edges representing influence relationships. We denote $n = |V|$ and $m = |E|$. In the triggering model, every node v has a distribution D_v over all subsets of its in-neighbors. Each node is either inactive or active, and once active, it stays active. Before the propagation starts, each node v samples a triggering set $T_v \sim D_v$. The propagation proceeds in discrete time steps $t = 0, 1, 2, \ldots$ At time $t = 0$, nodes in a given seed set $S \subseteq V$ are activated. For any time $t \geq 1$, an inactive node v becomes active if any only if at least one of its in-neighbors in T_v becomes active by time $t - 1$. The propagation ends when there are no newly activated nodes at a step. Two classical models, IC and LT, are both special cases of the triggering model. In the IC model, each edge (u, v) has an influence probability $p(u, v)$, and the triggering set T_v is sampled by independently sample every incoming edge (u, v) of v with success probability $p(u, v)$ and put u into T_v if the edge sample is successful; in the LT model, each edge (u, v) has an influence weight $w(u, v) \in [0, 1]$ such that $\sum_u w(u, v) \leq 1$, and at most, one in-neighbor u is sampled into T_v with probability proportional to $w(u, v)$. When considering time complexity, we assume that each sample T_v can be drawn with time proportional to the in-degree of v, and this holds for both IC and LT models.

A key quantity is the influence spread of a seed set S, denoted as $\sigma(S)$, which is defined as the expected number of final active nodes for the propagation starting from S. The classical influence maximization task is to select at most k seed nodes to maximize the influence spread, i.e., to find $S^* = \arg\max_{S \subseteq V,\ |S| \leq k} \sigma(S)$. The problem is NP-hard, and [1] proposed the greedy approximation algorithm, which is based on the submodularity of $\sigma(S)$ and guarantees $1 - 1/e - \varepsilon$ approximation for any small $\varepsilon > 0$.

In this article, we study the extension of influence maximization with general marketing strategies [1]. A mix of marketing strategies is modeled as a d-dimensional vector $\mathbf{x} = (x_1, \ldots, x_d) \in \mathbb{R}_+^d$, where $\mathbb{R}_+$ is the set of nonnegative real numbers. Each dimension corresponds to a particular marketing strategy, e.g., direct mail to one segment of the user base. Under the marketing strategy mix $\mathbf{x}$, each node $u \in V$ is independently activated as a seed with the probability given by the strategy activation function $h_u(\mathbf{x})$. Then, the set of activated seed nodes propagate the influence in the network following the triggering model. We define the influence spread of a marketing strategy mix $\mathbf{x}$ as the expected number of nodes activated and denote it as $g(\mathbf{x})$

$$g(\mathbf{x}) = \mathbb{E}_S[\sigma(S)] = \sum_{S \subseteq V} \sigma(S) \cdot \prod_{u \in S} h_u(\mathbf{x}) \cdot \prod_{v \notin S} (1 - h_v(\mathbf{x})). \quad (1)$$

The abovementioned formula can be interpreted as follows: for each subset of nodes S, under the marketing strategy mix $\mathbf{x}$, the probability that exactly nodes in S are activated as seeds and nodes not in S are not activated as seeds is given as $\prod_{u \in S} h_u(\mathbf{x}) \cdot \prod_{v \notin S} (1 - h_v(\mathbf{x}))$, which is because the node activations are independent. Then, given that exactly nodes in S are activated as seeds, the influence spread it generates is $\sigma(S)$. Therefore, enumerating through all possible subset set S, we obtain the abovementioned formula.

We further consider a class of functions where each strategy j independently tries to activate v as a seed. We refer to this case as independent strategy activation. Suppose that the set of strategies that may activate v is $S_v \subseteq [d]$, and the probability that strategy j with amount x_j activates v as a seed is $q_{v,j}(x_j)$, with $q_{v,j}(0) = 0$. Then, we have

$$h_v(\mathbf{x}) = 1 - \prod_{j \in S_v} \big(1 - q_{v,j}(x_j)\big). \quad (2)$$

Many application scenarios fit into the independent strategy activation assumption [see (2)]. We now give three such examples. The first application scenario is user segment marketing, in which each strategy j targets at a disjoint subset of users V_j. In this case, for each user v, it has a unique strategy targeted at v, i.e., $|S_v| = 1$.

The second scenario is personalized marketing, where each user is targeted with a personalized strategy. The personalized discount strategies studied in [16] belong to this scenario. Technically, this scenario is a special case of the abovementioned segment marketing scenario, where the user segments V_j's are all singletons, and $d = n$.

The third scenario is repeated marketing, such as multievent marketing. For example, each strategy j is a type of events, and x_j is the number of events of type j. Suppose that for each event of type j, a user v targeted by this event has an independent probability $r_{v,j}$ to be activated as a seed, and then, $q_{v,j}(x_j) = 1 - (1 - r_{v,j})^{x_j}$.

In this article, we consider discretized marketing strategies with granularity parameter δ, i.e., each strategy x_i takes discretized values $0, \delta, 2\delta, \ldots$ This set of vectors is referred to as a lattice and is denoted as $\mathcal{X}$. We consider the marketing strategy mix $\mathbf{x}$ with a total budget constraint k: $|\mathbf{x}| \le k$, where $|\mathbf{x}| = \sum_{i\in[d]} x_i$. The abovementioned constraint can be thought of as the total monetary budget constraint, where x_j is the monetary expense on strategy j, but other interpretations are also possible. Since we are doing influence maximization on lattice $\mathcal{X}$, we call it LIM, as formally defined in the following.

Definition 1 (LIM): Given a social network $G = (V, E)$ with the triggering model parameters $\{D_v\}_{v\in V}$ and the strategy activation functions $\{h_v\}_{v\in V}$ and a total budget k, the task of LIM, denoted as LIM, is to find an optimal strategy mix $\mathbf{x}^*$ that achieves the largest influence spread within the budget constraint, that is

$$\mathbf{x}^* = \operatorname*{argmax}_{\mathbf{x}\in\mathcal{X}, |\mathbf{x}|\le k} g(\mathbf{x}).$$

Note that if $\mathcal{X} = \{0, 1\}^n$ and $h_v(\mathbf{x}) = x_v$, that is, v is activated as a seed; if and only if it is selected by strategy $\mathbf{x}$, the LIM problem becomes the classical influence maximization problem. Therefore, LIM is more general and inherits the NP-hardness of the classical problem. For convenience, we sometimes also use LIM to refer to the lattice-based propagation model described earlier.

To solve the LIM problem, Kempe *et al.* [1] proposed the greedy algorithm based on the diminishing return property of $g(\mathbf{x})$, commonly referred to as the DR-submodular property [26], [29]. For two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, we denote $\mathbf{x} \le \mathbf{y}$ if $x_i \le y_i$ for all $i \in [d]$. Let $\mathbf{e}_i \in \mathbb{R}^d$ be the unit vector with the ith dimension being 1 and all other dimensions being 0. For a vector function $f : \mathcal{X} \to \mathbb{R}$, we say that f is DR-submodular if for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}$ with $\mathbf{x} \le \mathbf{y}$, for all $i \in [d]$, $f(\mathbf{x} + \delta\mathbf{e}_i) - f(\mathbf{x}) \ge f(\mathbf{y} + \delta\mathbf{e}_i) - f(\mathbf{y})$; we say that f is monotone (nondecreasing) if for all $\mathbf{x} \le \mathbf{y}$, $f(\mathbf{x}) \le f(\mathbf{y})$. Note that a set function f is monotone if $f(S) \le f(T)$ for all $S \subseteq T$ and submodular if $f(S\cup\{u\}) - f(S) \ge f(T\cup\{u\}) - f(T)$ for all $S \subseteq T$ and $u \notin T$. It is clear that if we represent sets as binary vectors and take step size $\delta = 1$, then it coincides with monotonicity and DR-submodularity of vector functions.

Algorithm 1 Algorithm L-Greedy(f, k, δ)

Input: monotone DR-submodular f, budget k, granularity δ
Output: vector $\mathbf{x}$

```
x = 0
for t = 1, 2, ..., k · δ^{-1} do
  j* = argmax_{j∈[d]} f(x + δe_j)
  x = x + δe_{j*}
end for
return x
```

When the vector function f on lattice $\mathcal{X}$ is nonnegative, monotone, and DR-submodular, the lattice-greedy (denoted as L-Greedy) algorithm as given in Algorithm 1 achieves $1 - 1/e$ approximation [30]. The L-Greedy algorithm searches the coordinate that gives the largest marginal return and moves one step of size δ on that coordinate, until it exhausts the budget.

To apply the L-Greedy algorithm to LIM, Kempe *et al.* [1] showed that when h_v's are monotone and DR-submodular with $\sigma(S)$ being monotone and submodular, the influence spread $g(\mathbf{x})$ given in (1) is also monotone and DR-submodular. Therefore, the L-Greedy algorithm can be applied to $g(\mathbf{x})$. As it is #P-hard to compute the influence spread $\sigma(S)$ in the IC and LT models [8], [19], we could use Monte Carlo simulations to estimate $g(\mathbf{x})$ to achieve $1 - 1/e - \varepsilon$ approximation for any small $\varepsilon > 0$.

We remark that in the LIM problem, for each strategy i, we can add an upper bound constraint $x_i \le b_i$ without changing the problem because we can extend the domain of x_i beyond b_i by restricting $h_v(\mathbf{x})$ with some $x_i > b_i$ to be the value at the boundary $x_i = b_i$. It is easy to verify that this extension will neither affect monotonicity and DR-submodularity of function h_v nor will it affect the lattice-greedy algorithm.

III. Scalable Algorithms for LIM

It is well known that the Monte Carlo greedy algorithm is not scalable. In this article, we propose scalable solutions to the LIM problem based on the seminal RIS approach [9]–[11], [21], [22]. In particular, we adapt the IMM algorithm of [11] in two different ways: one relies on partial RR sets and is denoted as IMM-PRR, and the other uses VSNs and is denoted as IMM-VSN.

These two algorithms present two complementary approaches for extending the RIS approach to the LIM setting: IMM-PRR extends the connection between the influence spread function and the RR set (as shown in Lemma 1), while IMM-VSN reduces the LIM problem into a standard influence maximization by adding virtual nodes in the graph to represent strategies. We want to evaluate both approaches, and thus, we present detailed implementation of both approaches even though our theoretical and experimental results indicate that IMM-VSN provides better efficiency, at least in the case of independent strategy activation.

We remark that the IMM algorithm shares the same algorithm structure with other later RIS-based algorithms, such as DSSA [21] and OPIM [22]. Thus, these algorithms can also

be incorporated into our algorithms. We choose IMM as our base algorithm mostly because it is relatively easy to present and analyze.

A. RR Sets and Its Properties

The RIS approach is based on the key concept of the RR sets, as defined in the following.

Definition 2 (RR Set): Under the triggering model, an RR set rooted at a node v, denoted R_v, is the random set of nodes v that reaches in one reverse propagation: sample all triggering sets $\{T_u\}_{u\in V}$, such that edges $\{(w,u) \mid u \in V, w \in T_u\}$ together with nodes V form a live-edge graph, and R_v is the set of nodes that can reach v (or v can reach reversely) in this live-edge graph. An RR set R without specifying a root is one with root v selected uniformly at random from V.

Intuitively, RR sets rooted at v store nodes that are likely to influence v. Technically, it has the following important connection with the influence spread of a seed set S: $\sigma(S) = n \cdot \mathbb{E}_R[\mathbb{I}\{S \cap R \neq \emptyset\}]$, where $\mathbb{I}$ is the indicator function [9], [11].

For the LIM problem, our solution space is no longer the sets of seed nodes, so the abovementioned connection between the influence spread $\sigma(S)$ and the RR set R cannot directly help us in efficient algorithm design. Fortunately, we could build a new connection between the influence spread $g(\mathbf{x})$ in the new solution space of strategy mix and the RR set R, as given in the following lemma. This connection enables us to extend the RIS approach to the LIM setting. Our first algorithm IMM-PRR is based on this general connection.

Lemma 1: For any strategy mix $\mathbf{x} \in \mathcal{X}$, we have

$$g(\mathbf{x}) = n \cdot \mathbb{E}_R\left[1 - \prod_{v\in R}(1 - h_v(\mathbf{x}))\right]. \tag{3}$$

Proof: By (1), we have $g(\mathbf{x}) = \mathbb{E}_S[\sigma(S)] = n \cdot \mathbb{E}_{S,R}[\mathbb{I}\{S \cap R \neq \emptyset\}] = n \cdot \mathbb{E}_R[\Pr_S\{S \cap R \neq \emptyset\}]$. Then, $\Pr_S\{S \cap R \neq \emptyset\}$ is the probability that at least one node in R (now fixed) is activated as a seed under strategy mix $\mathbf{x}$, so it is $1 - \prod_{v\in R}(1 - h_v(\mathbf{x}))$. □

Lemma 1 indicates that an RR set R is partially covered by a strategy mix $\mathbf{x}$ with probability (or weight) $1 - \prod_{v\in R}(1 - h_v(\mathbf{x}))$, instead of the classical case where an RR set is either fully covered by a seed set S or not. This lead to the partial RR set extension of IMM, called IMM-PRR.

B. Algorithm IMM-PRR

1) General Structure of IMM-PRR*:* By (3), we can generate θ independent RR sets as a collection $\mathcal{R}$ to obtain

$$\hat{g}_{\mathcal{R}}(\mathbf{x}) = \frac{n}{\theta}\sum_{R\in\mathcal{R}}\left(1 - \prod_{v\in R}(1 - h_v(\mathbf{x}))\right) \tag{4}$$

as an unbiased estimate of $g(\mathbf{x})$. Moreover, we have the following property for $\hat{g}_{\mathcal{R}}(\mathbf{x})$.

Lemma 2: lemhtogsubmodular If h_v is monotone and DR-submodular for all $v \in V$, then functions g and $\hat{g}_{\mathcal{R}}$ are also monotone and DR-submodular.

Proof (Sketch): We apply the technical Lemma 3 below on $\prod_{v\in R}(1 - h_v(\mathbf{x}))$, and notice that $1 - h_v(\mathbf{x})$ is nonnegative, monotone nonincreasing, and DR-supermodular. Therefore, $1 - \prod_{v\in R}(1 - h_v(\mathbf{x}))$ is nonnegative, monotone increasing, and DR-submodular. □

Algorithm 2 General Structure of IMM-PRR

[1]

Input: G: the social graph; $\{D_v\}_{v\in V}$: triggering model parameters; $\{h_v\}_{v\in V}$: strategy activation functions (or $\{q_{v,j}\}_{v\in V,\ j\in S_v}$ for L-GreedyDelta;) k: budget; δ: granularity; ε: accuracy; ℓ: confidence

Output: $\mathbf{x} \in \mathcal{X}$

$\mathcal{R} = \text{Sampling}(G, \{D_v\}_{v\in V}, k, \delta, \varepsilon, \ell)$

$\mathbf{x} = \text{L-Greedy}(\hat{g}_{\mathcal{R}}, k, \delta)$

// or $\text{L-GreedyDelta}(\mathcal{R}, \{q_{v,j}\}_{v\in V,\ j\in S_v}, k, \delta)$

return x

Lemma 3: If f_1 and f_2 are nonnegative, monotone nonincreasing, and DR-supermodular, then $f(\mathbf{x}) = f_1(\mathbf{x})f_2(\mathbf{x})$ is also monotone nonincreasing and DR-supermodular.

Proof: The monotonicity is straightforward. For DR-supermodularity, for any $\mathbf{x} \leq \mathbf{y}$, we have

$$\begin{aligned}
&(f(\mathbf{x}+\delta\mathbf{e}_i) - f(\mathbf{x})) - (f(\mathbf{y}+\delta\mathbf{e}_i) - f(\mathbf{y}))\\
&\quad= f_1(\mathbf{x}+\delta\mathbf{e}_i)f_2(\mathbf{x}+\delta\mathbf{e}_i) - f_1(\mathbf{x})f_2(\mathbf{x})\\
&\qquad- (f_1(\mathbf{y}+\delta\mathbf{e}_i)f_2(\mathbf{y}+\delta\mathbf{e}_i) - f_1(\mathbf{y})f_2(\mathbf{y}))\\
&\quad= f_1(\mathbf{x}+\delta\mathbf{e}_i)(f_2(\mathbf{x}+\delta\mathbf{e}_i) - f_2(\mathbf{x}))\\
&\qquad+ f_2(\mathbf{x})(f_1(\mathbf{x}+\delta\mathbf{e}_i) - f_1(\mathbf{x}))\\
&\qquad- f_1(\mathbf{y}+\delta\mathbf{e}_i)(f_2(\mathbf{y}+\delta\mathbf{e}_i) - f_2(\mathbf{y}))\\
&\qquad- f_2(\mathbf{y})(f_1(\mathbf{y}+\delta\mathbf{e}_i) - f_1(\mathbf{y}))\\
&\quad\leq f_1(\mathbf{x}+\delta\mathbf{e}_i)(f_2(\mathbf{y}+\delta\mathbf{e}_i) - f_2(\mathbf{y}))\\
&\qquad+ f_2(\mathbf{x})(f_1(\mathbf{y}+\delta\mathbf{e}_i) - f_1(\mathbf{y}))\\
&\qquad- f_1(\mathbf{y}+\delta\mathbf{e}_i)(f_2(\mathbf{y}+\delta\mathbf{e}_i) - f_2(\mathbf{y}))\\
&\qquad- f_2(\mathbf{y})(f_1(\mathbf{y}+\delta\mathbf{e}_i) - f_1(\mathbf{y}))\\
&\quad= (f_1(\mathbf{x}+\delta\mathbf{e}_i) - f_1(\mathbf{y}+\delta\mathbf{e}_i))(f_2(\mathbf{y}+\delta\mathbf{e}_i) - f_2(\mathbf{y}))\\
&\qquad+ (f_2(\mathbf{x}) - f_2(\mathbf{y}))(f_1(\mathbf{y}+\delta\mathbf{e}_i) - f_1(\mathbf{y})) \leq 0
\end{aligned}$$

where the first inequality is due to the DR-supermodular and nonnegative conditions, and the second inequality is due to the monotone nonincreasing property. □

With Lemma 2, we can apply the L-Greedy algorithm on $\hat{g}_{\mathcal{R}}$. Let $\hat{\mathbf{x}}^o = \text{L-Greedy}(\hat{g}_{\mathcal{R}}, k, \delta)$. When $\theta = |\mathcal{R}|$ is large enough, $\hat{g}_{\mathcal{R}}$ is very close to g, and we could show that $\hat{\mathbf{x}}^o$ is a $1 - 1/e - \varepsilon$ approximation for the LIM problem.

This leads to the general structure of the IMM-PRR algorithm, as given in Algorithm 2, similar to the IMM algorithm. The algorithm takes the input as listed in Algorithm 2 and outputs $\mathbf{x}$ such that $\mathbf{x}$ is a $1-1/e-\varepsilon$ approximate solution to the LIM problem with probability at least $1 - n^{\ell}$. The algorithm contains two phases. In the first phase, the Sampling procedure determines the number of RR sets needed and generates these RR sets; in the second phase, a lattice-greedy algorithm on these RR sets is used to find the resulting strategy vector $\mathbf{x}$. We first discuss the second phase, which requires major changes from the original IMM algorithm, and then introduce the first phase.

2) Efficient L-Greedy *on RR Sets Under Independent Strategy Activation:* If the strategy activation function $h_v(\cdot)$'s are given as black boxes, we have to compute $h_v(\mathbf{x})$ from scratch. Suppose that the running time cost for computing $h_v(\mathbf{x})$ is $O(T_{h_v})$. Then, it is straightforward to verify that the L-Greedy$(\hat{g}, k, \delta)$ algorithm with the computation of $\hat{g}(\mathbf{x})$, as given in (4), has time complexity $O(k \cdot \delta^{-1} \cdot d \cdot \sum_{R\in\mathcal{R}} \sum_{v\in R} T_{h_v})$.

When we have further structural knowledge about h_v's, we can greatly improve the efficiency of the L-Greedy algorithm. In particular, we consider the case of independent strategy activation, as given in (2). That is, each individual strategy j independently tries to activate node v with probability of success $q_{v,j}(x_j)$. We assume that function $q_{v,j}(x)$ is nondecreasing and concave for every node v and every strategy $j \in S_v$. Recall that in the event marketing scenario, we have $q_{v,j}(x_j) = 1 - (1 - r_{v,j})^{x_j}$, and thus, in this case, indeed, $q_{v,j}(x)$ is nondecreasing and concave. The following lemma shows that when $q_{v,j}(x)$ is nondecreasing and concave, $h_v(\mathbf{x})$ is monotone and DR-submodular.

Lemma 4: lemDRSubmodularqtoh If function $q_{v,j}(x)$ is nondecreasing and concave for every $j \in S_v$, then $h_v(\mathbf{x})$ is monotone and DR-submodular.

Proof (Sketch): The proof also uses Lemma 3, and we only need to notice that 1-D convexity is a special case of DR-supermodularity. □

Equation (2) enables more efficient updates for L-Greedy. Instead of always computing $\hat{g}_{\mathcal{R}}(\mathbf{x} + \delta\mathbf{e}_j)$ from scratch in L-Greedy$(\hat{g}, k, \delta)$, we compute $\Delta_j(\mathbf{x}) = \hat{g}_{\mathcal{R}}(\mathbf{x}+\delta\mathbf{e}_j) - \hat{g}_{\mathcal{R}}(\mathbf{x})$, which is given in the following equation:

$$\Delta_j(\mathbf{x}) = \frac{n}{\theta} \sum_{R\in\mathcal{R}} \left(\prod_{v\in R} \prod_{j'\in S_v} \left(1 - q_{v,j'}\left(x_{j'}\right)\right) \right) \cdot \left(1 - \frac{\prod_{v:v\in R, j\in S_v} \left(1 - q_{v,j}\left(x_j + \delta\right)\right)}{\prod_{v:v\in R, j\in S_v} \left(1 - q_{v,j}\left(x_j\right)\right)} \right). \quad (5)$$

The advantage of (5) is in reusing past computations. Specifically, the term within the first parentheses is the same across all strategies, so its computation can be shared. Moreover, since it is often the case that each user is only exposed to a small subset of strategies (i.e., $|S_v|$ is smaller than d), we carefully maintain a data structure to improve the efficiency when $|S_v| < d$. Algorithm 3 presents the detailed lattice-greedy update procedure L-GreedyDelta, which replaces L-Greedy$(\hat{g}, k, \delta)$ when (2) holds.

In Algorithm 3, we use s_i to store the term $\prod_{v\in R_i} \prod_{j'\in S_v} (1 - q_{v,j'}(x_{j'}))$ in (5) shared across different strategies j. We use ratio to store the ratio term $\prod_{v:v\in R, i\in S_v} (1 - q_{v,i}(x_i + \delta))(1 - q_{v,i}(x_i))^{-1}$ in (5). List$_j$ is a linked list for strategy j, and it stores the pair (i, v), which means that RR set R_i contains node v that can be affected by strategy j. The list is ordered by RR set index i first and then by node index v. In each round t, the algorithm iterates through all strategies j (lines 7–20) to compute $\Delta_j(\mathbf{x})$ for the current $\mathbf{x}$. In particular, for each strategy j, the algorithm traverses the List$_j$ (lines 9–16); for the segment with the same RR set index i, it updates ratio; and when it reaches a new RR set index ($i \neq$ prev), it cumulates Δ_j, as given in (5), for the corresponding RR set. The reason we maintain List$_j$ of pairs instead of simply looping through all RR set indices i and then all nodes within R_i is that RR sets are usually not very large, and it is likely that no node in RR set R_i is affected by strategy j and, thus, not looping through all RR sets save time. After computing $\Delta_j = \Delta_j(\mathbf{x})$, we find the strategy j^* with the largest Δ_j (line 21), move along the direction of j^* for one step (line 22), and then update all shared terms s_i's (line 23).

Algorithm 3 L-GreedyDelta: Efficient Lattice-Greedy Implementation on RR Sets

Input: $\mathcal{R} = \{R_1, \ldots, R_\theta\}$: RR sets; $\{q_{v,j}\}_{v\in V,\ j\in S_v}$; k: budget; δ: granularity
Output: $\mathbf{x} \in \mathcal{X}$

```
x = (x_1, ··· , x_d) = 0
// Lines 3–5 can be done while generating RR sets
s = (s_0, s_1, ··· , s_θ) with s_0 = 0, s_i = ∏_{v∈R_i} ∏_{j∈S_v} (1 − q_{v,j}(x_j))
∀j ∈ [d], List_j = ∅
∀R_i ∈ R, ∀v ∈ R_i, ∀j ∈ S_v, append (i, v) to List_j
for t = 1, 2, ··· , k · δ^{-1} do
  for j ∈ [d] do
    Δ_j = 0, prev = 0, ratio = 1
    for (i, v) ∈ List_j do
      if i ≠ prev then
        Δ_j = Δ_j + s_prev · (1 − ratio)
        ratio = 1
        prev = i
      end if
      ratio = ratio · (1−q_{v,j}(x_j+δ)) / (1−q_{v,j}(x_j))
    end for
    if prev ≠ 0 then
      Δ_j = Δ_j + s_prev · (1 − ratio)
    end if
  end for
  j* = argmax_{j∈[d]} Δ_j
  x = x + δe_{j*}
  ∀i ∈ [θ], s_i = s_i · ∏_{v∈R_i : j*∈S_v} (1 − q_{v,j*}(x_{j*} + δ)) · (1 − q_{v,j*}(x_{j*}))^{-1}
end for
return x
```

Suppose that the running time cost for computing each $q_{v,j}(x_j)$ is a constant. Then, we have Lemma 5.

Lemma 5: lemtimeDelta The time complexity of L-GreedyDelta is $O(k \cdot \delta^{-1} \cdot (\sum_{R\in\mathcal{R}} \sum_{v\in R} |S_v|))$.

Proof of Lemma 5 (Sketch): The algorithm has totally $k\delta^{-1}$ rounds. In each round, it enumerates all tuples (i, v, j) for RR set R_i, node $v \in R_i$, and strategy $j \in S_v$, and for each tuple, it has a constant number of calls to function $q_{v,j}$, so the running time in one round t is $O(\sum_{R\in\mathcal{R}} \sum_{v\in R} |S_v|)$. □

Notice that if we compute $\hat{g}(\mathbf{x} + \delta\mathbf{e}_j)$ directly instead of $\Delta_j(\mathbf{x})$, we have $T_{h_v} = O(d)$. Then, the time complexity is $O(k \cdot \delta^{-1} \cdot d \cdot \sum_{R\in\mathcal{R}} \sum_{v\in R} |S_v|)$, which is worse than L-GreedyDelta by a factor of d.

3) First Phase Sampling *Procedure:* The Sampling procedure in the first phase is to generate enough RR sets $\mathcal{R}$ to provide the theoretical guarantee on the approximation ratio. It is a minor variation of the Sampling procedure of IMM in [11]. In particular, they show that the number of RR sets $\theta = \Theta(n \log n/\text{OPT})$ is enough, where OPT is the optimal solution. They estimate a lower bound LB of OPT by iteratively guessing $n/2, n/4, n/8, \ldots$ as lower bounds and using the greedy procedure on obtained RR sets to verify if the guess is correct. We use the same procedure, with only two differences: 1) we use L-GreedyDelta procedure to replace the greedy procedure on RR sets and 2) we replace $\ln \binom{n}{k}$ with $\min(k\delta^{-1} \ln d, d \ln(k\delta^{-1}))$ in the two parameters, λ' and λ^*, because both $d^{k\delta^{-1}}$ and $(k\delta^{-1})^d$ are upper bounds on the number of vectors satisfying the constraint $|\mathbf{x}| \leq k$. The bound $d^{k\delta^{-1}}$ is because we have $k\delta^{-1}$ greedy steps, and each step selects one dimension among d dimensions, and the bound $(k\delta^{-1})^d$ is because each dimension has at most $k\delta^{-1}$ choices, and we have d dimensions combined together. We can see that when d is large (e.g., personalized marketing with $d = n$) but $k\delta^{-1}$ is relatively small (coarse granularity), we would use $k\delta^{-1} \ln d$, but when $k\delta^{-1}$ is large (fine granularity) but d is small (e.g., only a few global strategies), we could use $d \ln(k\delta^{-1})$. Henceforth, we let $M = \min(k\delta^{-1} \ln d, d \ln(k\delta^{-1}))$. The pseudocode for the Sampling procedure is included in Algorithm 4, with parameter $\lambda^*(\ell)$ defined as follows:

$$\lambda^*(\ell) = 2n \cdot ((1 - 1/e) \cdot \alpha + \beta)^2 \cdot \varepsilon^{-2}$$
$$\alpha = \sqrt{\ell \ln n + \ln 2}, \quad \beta = \sqrt{(1 - 1/e) \cdot \left(M + \alpha^2\right)}. \quad (6)$$

We remark that Chen [31] pointed out an issue in the original IMM algorithm and provided two workarounds, and we adopt the more efficient workaround 2 (lines 2–3). Algorithms 2–4 form the IMM-PRR algorithm. The following theorem summarizes the theoretical guarantee of the IMM-PRR algorithm.

Theorem 1: Under the case of independent strategy activation [see (2)], the IMM-PRR algorithm returns a $(1 - 1/e - \varepsilon)$-approximate solution to the LIM problem with at least $1 - 1/n^\ell$ probability. When $q_{v,j}$'s are such that the optimal solution of LIM is at least as good as the best single node influence spread, IMM-PRR runs in $O(k\delta^{-1}(\max_{v \in V} |S_v|)(M + \ell \log n)(n + m)/\varepsilon^2)$ expected time, where $M = \min(k\delta^{-1} \ln d, d \ln(k\delta^{-1}))$.

The proof of the theorem mainly follows the analysis of IMM in [11], and the novel part of the analysis is already mostly shown in the previous lemmas. The remaining part of the proof is given in the Appendix. Note that the abovementioned technical assumption that assuming the optimal solution is at least as good as the best single node influence spread is reasonable since it means the budget and the functions $q_{v,j}$'s are at least good enough to activate one single best node. If it is not true, the entire marketing scheme is not very useful anyway. Comparing with the time complexity $O((k + \ell)(m + n) \log n/\varepsilon^2)$ of IMM in [11], the main added difficulty is that a strategy can only partially cover an RR set (see Lemma 1), which implies that, in each greedy step, we have to process all RR sets. We will overcome this issue by an alternative reduction approach in Section III-C.

Algorithm 4 First Phase Sampling Procedure

Input: G: the social graph; $\{D_v\}_{v \in V}$: triggering model parameters; $\{q_{v,j}\}_{v \in V,\ j \in S_v}$: strategy-node activation functions; k: budget, δ: granularity; ε: accuracy; ℓ: confidence
Output: A collection of RR sets $\mathcal{R}$

```
R = ∅; LB = 1
compute γ via binary search such that ⌈λ*(ℓ+γ)⌉/n^{ℓ+γ} ≤ 1/n^ℓ // workaround 2 in [31], with λ*(ℓ) defined in Eq. (6)
ℓ = ℓ + γ + ln 2/ln n
Let ε' = √2 · ε
for i = 1, 2, ··· , log_2 n do
  Let y = n/2^i
  θ_i = λ'/y, where λ' = ((2+(2/3)ε')·(M+ℓ·ln n+ln log_2 n)·n)/ε'^2.
  while |R| ≤ θ_i do
    Select a node v from G uniformly at random
    Generate an RR set for v, and insert it into R
  end while
  x = L-GreedyDelta(R, {q_{v,j}}_{v∈V, j∈S_v}, k, δ)
  if ĝ_R(x) ≥ (1 + ε') · y then
    LB = ĝ_R(x)/(1 + ε')
    break
  end if
end for
θ = λ*(ℓ)/LB, where λ*(ℓ) is defined in Eq. (6)
while |R| ≤ θ do
  Select a node v from G uniformly at random
  Generate an RR set for v, and insert it into R
end while
return R
```

C. Algorithm IMM-VSN for Independent Strategy Activation

In this section, we consider an alternative design choice under independent strategy activation. The idea is that since each strategy independently activates nodes, we may be able to introduce virtual nodes representing strategies such that the LIM model is reduced to the classical triggering model, and then, we could apply algorithms, such as IMM, to solve the classical influence maximization problem under the reduced model. It turns out that we need to incorporate a mixture of LT and IC models for the interaction between the virtual nodes and the real nodes and carefully argue about the equivalence between LIM and the reduced model. We refer this new algorithm as IMM-VSN (VSN stands for virtual strategy node).

In IMM-VSN, for every strategy j, we construct VSN set $U_j = \{u_{j,1}, u_{j,2}, \ldots, u_{j,k\delta^{-1}}\}$, and for every real node v in the original graph and every strategy $j \in S_v$, we connect every virtual node $u_{j,i}$ to v with a directed virtual edge $(u_{j,i}, v)$. Let $U = \bigcup_{j=1}^{d} U_j$ be the set of all virtual nodes. The purpose is such that the prefix set $U_{j,i} = \{u_{j,1}, \ldots, u_{j,i}\}$ corresponds to the quantity $x_j = i\delta$ for strategy j, and if nodes in $U_{j,i}$ are seeds, then the real node v is activated with probability $q_{v,j}(i\delta)$, the probability that amount $i\delta$ of strategy j would activate v [see (2)]. To do so, we utilize the LT model

as follows. For each edge $(u_{j,i}, v)$, we assign LT weight

$$w(u_{j,i}, v) = q_{v,j}(i\delta) - q_{v,j}((i-1)\delta). \tag{7}$$

When a seed set $S \subseteq U$ of virtual nodes attempts to activate a real node v, we first consider seed set within each strategy $S \cap U_j$, and nodes in $S \cap U_j$ attempt to activate v following the LT model with weights defined in (7). Then, among different strategies, their attempts to activate v are independent, and v is activated as long as seeds from one strategy activates v. This is a mixture of IC and LT models and is our key to allow the reduction to work.

We denote the augmented graph together with the above-described propagation model as G_A. In G_A, only virtual nodes can be selected as seeds, and only real nodes are counted toward the influence spread. The propagation in G_A starts from the seeds in the VSNs, and these seeds activate real nodes according to the abovementioned IC and LT mixture models. Then, the propagation among real nodes follows the original triggering model. The reason that this reduction works is justified by the following theorem.

Theorem 2: Under the independent strategy activation model [see (2)], the following holds: 1) for any strategy mix $\mathbf{x} = (x_1, \ldots, x_d) \in \mathcal{X}$, the distribution of the set of nodes activated by $\mathbf{x}$ in the LIM model is the same as the distribution of the set of real nodes activated by seed set $S^{\mathbf{x}} = \bigcup_{j=1}^{d} U_{j,x_j\delta^{-1}}$ in G_A and 2) conversely, for any seed set $S \subseteq U$, we can map S to $\mathbf{x}^S = (x_1^S, \ldots, x_d^S)$ where $x_j^S = |S \cap U_j| \cdot \delta$, such that the influence spread of S in G_A (only counting the activation of the real nodes) is at most the influence spread of $\mathbf{x}^S$ in the LIM model. As a consequence, if an approximation algorithm for the triggering model produces S on graph G_A, then $\mathbf{x}^S$ would be an approximate solution for LIM with the same approximation ratio.

Proof: First, given strategy mix $\mathbf{x}$, by the LT model and our weight construction [see (7)], we know that the probability that the seed set $S^{\mathbf{x}} \cap U_j = U_{j,x_j\delta^{-1}}$ activates node v in G_A is $\sum_{i=1}^{x_j\delta^{-1}} w(u_{j,i}, v) = q_{v,j}(x_j)$, which coincides with the probability that strategy j with amount x_j would activate v in the LIM model. Among different strategy seed nodes, they attempt to activate v independently, which coincides with (2) that governs the activation of v from strategy $\mathbf{x}$. Since the remaining propagation among real nodes follows the same model, we can conclude that the set of nodes activated in either the LIM model or G_A follows the same distribution.

Conversely, let S be a seed set in G_A. For each strategy j, $S \cap U_j$ may not be the prefix set. Let U_{j,x_j^S} be the corresponding prefix set with $x_j^S = |S \cap U_j|$. We claim that U_{j,x_j^S} activates v with probability at least as high as that of $S \cap U_j$ activating v. Here, we need to critically use the concaveness of $q_{v,j}$: by its concaveness, we know that edge weight $w(u_{j,i}, v)$ is nonincreasing over i. Then, the sum of weights of the prefix set U_{j,x_j^S} to v is at least as large as the sum of the weights of $S \cap U_j$ to v. Thus, by the LT model, our claim holds. Once the claim holds, we know that, by moving the seeds to the prefix, we always have a higher probability of activating each real node. By the first part of the proof, we know that the prefix seed set exactly corresponds to the strategy mix $\mathbf{x}^S = (x_1^S, \ldots, x_d^S)$. Therefore, the influence spread of $\mathbf{x}^S$ in the LIM model must be at least as high as the influence spread of S in G_A.

The final part of the approximation algorithm becomes straightforward, once we have the abovementioned results. □

We remark that part 2 of the theorem critically depends on the concaveness of $q_{v,j}$ and is where we need to use the LT model construction. We could use the IC model with proper edge probability assignment for part (1), but it appears that IC model would not allow us to use the concaveness of $q_{v,j}$ to show part 2. This is why we use a mixture of the IC and LT models in the end.

With Theorem 2, our algorithmic design for IMM-VSN is clear, and its general structure is summarized in Algorithm 5. We first construct the augmented graph G_A and then apply an existing algorithm, in our case IMM, on G_A to find a seed set S of virtual nodes with budget $k\delta^{-1}$, and finally, we convert S to $\mathbf{x}^S$, as specified in Theorem 2, as our solution. When using IMM, we also employ the following adaptations to improve its performance for the special G_A graph.

1) At each real node v when we want to generate one more step in the reverse simulation, we first sample v's triggering set $T_v \sim D_v$ and put nodes in T_v in the RR set, and these are real nodes; then, for each strategy $j \in S_v$, we randomly pick at most one virtual node $u_{j,i}$ with probability $w(u_{j,i}, v)$ following the LT model, and and this can be efficiently implemented by a binary search; finally, we do reverse simulation for each strategy j independently, which corresponds to the independent activation across different strategies.
2) Since only virtual nodes are seeds, an RR set without virtual nodes will be discarded, and greedy seed selection is only among the virtual nodes.

Algorithm 5 General Structure of Algorithm IMM-VSN

Input: G: the social graph; $\{D_v\}_{v \in V}$: triggering model parameters; $\{q_{v,j}\}_{v \in V,\ j \in S_v}$: strategy-node activation functions; k: budget; δ: granularity; ε: accuracy; ℓ: confidence

Output: $\mathbf{x} \in \mathcal{X}$

generate augmented graph G_A and the diffusion model on it as follows: (1) add VSNs $U = \bigcup_{j=1}^{d} U_j$ to the node set, where $U_j = \{u_{j,1}, u_{j,2}, \ldots, u_{j,k\delta^{-1}}\}$; (2) add directed edges $\{(u_{j,i}, v) | v \in V,\ j \in S_v, u_{j,i} \in U_j\}$ to the edge set; (3) each edge $(u_{j,i}, v)$ has LT weight $w(u_{j,i}, v) = q_{v,j}(i\delta) - q_{v,j}((i-1)\delta)$; (4) triggering set distribution of every real node v is adjusted such that: (4.1) real nodes are selected by D_v; (4.2) virtual nodes in U_j with $j \in S_v$ are selected independent of real nodes and other virtual nodes; (4.3) within U_j, virtual node $u_{j,i}$ is selected exclusively with probability $w(u_{j,i}, v)$, just like in the LT model

run IMM on graph G_A with budget $k\delta^{-1}$ and obtain seed set $S \subseteq U$ on virtual nodes. IMM is adapted for G_A as described in the text

$\mathbf{x} = (x_1^S, \ldots, x_d^S)$ where $x_j^S = |S \cap U_j| \cdot \delta$

return x

3) Since only real nodes are counted toward the influence spread, we only uniformly at random pick roots of RR sets among real nodes.
4) By part 2 of Theorem 2, in the greedy NodeSelection procedure of IMM (corresponding to the L-Greedy procedure in IMM-PRR), after selecting all the seed nodes, we convert them to the prefix node set for each strategy.
5) The total number of possible strategy mixes is at most $M = \min(k\delta^{-1}\ln d, d\ln(k\delta^{-1}))$ as discussed in Section III-B, and together with part (d) above, we know that the total number of seed set outputs is also at most M; therefore, we will use M to replace $\binom{n}{k}$ in the original IMM algorithm.

The approximation guarantee of IMM-VSN is ensured by the correctness of the IMM algorithm plus Theorem 2. For time complexity, our adaptions to IMM save running time. Overall, we have Theorem 3.

Theorem 3: Under the case of independent strategy activation [see (2)], the IMM-VSN algorithm returns a $(1 - 1/e - \varepsilon)$-approximate solution to the LIM problem with at least $1 - 1/n^{\ell}$ probability. When $q_{v,j}$'s are such that the optimal solution of LIM is at least as good as the best single node influence spread, IMM-VSN runs in $O((M+\ell\log n)(m+\log(k\delta^{-1})\sum_{v\in V}|S_v|)/\varepsilon^2)$ expected time, where $M = \min(k\delta^{-1}\ln d, d\ln(k\delta^{-1}))$.

The proof of the theorem follows that of [11], and the novel part of the analysis is mainly summarized and proved in Theorem 2. The remaining part of the proof is given in the Appendix. Comparing the running time result of Theorem 3 with that of Theorem 1, we can see that the key difference is between the term $(m+\log(k\delta^{-1})\sum_{v\in V}|S_v|)$ of IMM-VSN and the term $k\delta^{-1}\max_{v\in V}|S_v|(m+n)$ of IMM-PRR. IMM-VSN seems to have a better running time especially in avoiding an extra term of $k\delta^{-1}$, which is partly because it does not require maintaining partial RR sets and partly because of the efficient LT reverse sampling method via binary search. Of course, these theoretical results are all upper bounds, so we cannot formally conclude the superiority of IMM-VSN. We will demonstrate the superior performance of IMM-VSN through our empirical evaluation. We also want to point out that IMM-VSN only works for the case of independent strategy activation, while IMM-PRR works for more general cases, and thus, we cannot say that IMM-VSN can always replace IMM-PRR.

IV. LIM With Partitioned Budgets

In this section, we further generalize the LIM problem with partitioned budgets. More specifically, marketing strategies often belong to multiple categories, and each category may be assigned a separate budget. Formally, the strategy set $[d]$ is partitioned into λ categories $C_1, \ldots, C_\lambda$, and each category C_j has a budget k_j, i.e., $\sum_{i\in C_j} x_i \le k_j$. For convenience, we use $\mathbf{x}_C$ to denote the projection of vector $\mathbf{x}$ into index set C. Then, the abovementioned constraint is $|\mathbf{x}_{C_j}| \le k_j$. The partitioned budget problem is formally defined in the following.

Definition 3 (LIM With Partitioned Budgets): Given the same input as in the LIM problem (see Definition 1), except that total budget k is replaced by partitions $\{C_j\}_{j\in[\lambda]}$ and partitioned budgets $\{k_j\}_{j\in[\lambda]}$, the task of LIM with partitioned budgets, denoted as LIM-PB, is to find an optimal strategy mix $\mathbf{x}^*$ that achieves the largest influence spread within the partitioned budget constraints, that is

$$\mathbf{x}^* = \operatorname*{argmax}_{\mathbf{x}\in\mathcal{X}, |\mathbf{x}_{C_j}|\le k_j, \forall j\in[\lambda]} g(\mathbf{x}).$$

Note that since the per-strategy constraint $x_i \le b_i$ for the original LIM problem does not change the problem, our partitioned constraint here means that $|C_j| > 1$ for all $j \in [d]$.

We, next, explain how to solve the partitioned budget constraint version LIM-PB. Our method relies on the submodular maximization problem under the general matroid constraint. A matroid on a set of elements U is a collection of subsets of U called independent sets that satisfy the following two properties: 1) if $I \subseteq U$ is an independent set, then every subset of I is also an independent set and 2) if I and I' are two independent sets with $|I| < |I'|$, then there must be some element $e \in I' \setminus I$ such that $I \cup \{e\}$ is also an independent set. The simplest matroid is the uniform matroid, where, for some parameter k, all subsets I with $|I| \le k$ are independent sets. Classical influence maximization essentially uses the uniform matroid constraint. A partition matroid is such that, for a certain partition of U into disjoint sets, $A_1, \ldots, A_\lambda$, and for parameters, $k_1, \ldots, k_\lambda$, all subsets $I \subseteq U$ satisfying $|I \cap A_i| \le k_i$ for all $i \in [\lambda]$ are independent sets. The classical result in [32] shows that the greedy algorithm on a general matroid could achieve $1/2$ approximation ratio for nonnegative monotone and submodular set functions.

Through Lemma 2, we already know that our objective functions $g(\mathbf{x})$ and $\hat{g}_{\mathcal{R}}(\mathbf{x})$ are nonnegative, monotone, and DR-submodular, but they are vector functions. We now show how to translate them into equivalent set functions and then show that the LIM-PB problem corresponds to a partitioned matroid constraint under the set representation. Let $b \ge \sum_{j\in[\lambda]} k_j \cdot \delta^{-1}$ be a large enough integer. Construct the set of elements $U = \{(j, s) \mid j \in [d], s \in [b]\}$. For any subset $A \subseteq U$, denote $A^{(j)} = A \cap \{(j, s) \mid s \in [b]\}$. We map A into a vector $\mathbf{x}^A = (x_1^A, \ldots, x_d^A)$ such that $x_j^A = |A^{(j)}|\cdot\delta$. Conversely, for every vector $\mathbf{x} \in \mathcal{X}$ satisfying the partitioned budget constraint, we map $\mathbf{x}$ to a set $A^{\mathbf{x}} = \{(j, s) \mid j \in [d], s\cdot\delta \le x_j\}$. For every vector function f, we define a set function f^U on U to be $f^U(A) = f(\mathbf{x}^A)$, for all $A \subseteq U$. It is easy to see that the marginal $f^U(A \cup \{(j, s)\}) - f^U(A) = f(\mathbf{x}^A + \delta\mathbf{e}_j) - f(\mathbf{x}^A)$. Thus, one can verify that if f is monotone and DR-submodular, then f^U is monotone and submodular. Next, for the partitioned budget constraint $|\mathbf{x}_{C_i}| \le k_i$ given partition $C_1, \ldots, C_\lambda$ of $[d]$ and budgets $k_1, \ldots, k_\lambda$, it is equivalent to partition U to $U_1, \ldots, U_\lambda$, with $U_i = C_i \times [b]$, and enforce constraint $|A \cap U_i| \le k_i \cdot \delta^{-1}$ for all $A \subseteq U$ and $i \in [\lambda]$. Therefore, we translate the LIM-PB problem of maximizing $g(\mathbf{x})$ with the partitioned budget constraint to maximizing $g^U(A)$ under the partition matroid constraint. Similarly, we can translate $\hat{g}(\mathbf{x})$ to $\hat{g}^U(A)$. Therefore, we can conclude that the greedy algorithms under the partitioned budget constraint could achieve $1/2$ approximation.

The actual greedy algorithm is straightforward. In IMM-PRR, in every greedy step, when we need to find another

TABLE I
DATA SET STATISTICS

Network	n	m	Average Degree
DM	679	3,374	4.96
NetHEPT	15,233	62,752	4.12
Flixster	29,357	425,228	14.48
DBLP	654,628	3,980,318	6.08
Orkut	3,072,441	117,185,083	78

increment in one of the strategies (line 3 of Algorithm 1 or line 21 of Algorithm 3), instead of taking argmax among all possible $j \in [d]$, we only search for j such that $\mathbf{x} + \delta \mathbf{e}_j$ still satisfies the partitioned budget constraint. Similarly, in IMM-VSN, when we need to find another VSN as a seed, we need to only search for those seeds that would satisfy the partitioned budget constraint. The greedy steps terminate until the partitioned budgets are exhausted. The corresponding algorithms achieve $1/2-\varepsilon$ approximation ratio with probability at least $1-1/n^{\ell}$ and run in the same expected running time as in their nonpartitioned versions.

V. EXPERIMENTS

A. Experiment Setup

1) Data Sets: We ran our experiments on five real-world networks, with statistics summarized in Table I. Three of them, denoted DM, NetHEPT, and DBLP, are collaboration networks: every node is an author, and every edge means the two authors collaborated on a paper. DM network is a network of data mining researchers extracted from the ArnetMiner archive (arnetminer.org) [33], and NetHEPT is a network extracted from the high-energy-physics section of arxiv.org, while DBLP is extracted from the computer science bibliography database dblp.org [8]. Their sizes are small (679 nodes), medium (15k nodes), and large (654k) nodes, respectively. We include the small DM data set mainly to suit the slow Monte Carlo greedy algorithm. The data set Flixster is a user network of the movie rating site flixster.com. Every node is a user and a directed edge from u to v means that v has rated some movie(s) that is u rated earlier [34]. The largest data set Orkut [35] is a social network extracted from the friendship network of users in Orkut. The IC model parameters of NetHEPT, DBLP, and Orkut are synthetically set using the weighted cascade method [1]: edge $p(u, v) = 1/d_v$, where d_v is the in-degree of node v. For the DM and Flixster networks, we obtain learned edge parameters from Tang *et al.* [33] and Barbieri *et al.* [34], respectively.

2) Application Scenarios: We test two application scenarios of independent strategy activation explained in Section II. The first is the personalized marketing scenario tested in [16]. In this scenario, each user v has one unique strategy x_v, such as the personalized discount to v, and $h_v(\mathbf{x})$ only depends on x_v. We set $h_v(\mathbf{x}) = 2\, x_v - x_v^2$ following the same setting in [16]. The second one is the segmented event marketing scenario, which is not covered by previous studies. In this case, each strategy j is targeting at a disjoint subset of users V_j, and x_j is the number of marketing events for user group V_j. In our experiments, we set $d = 200$ for each data set. Moreover, we choose top $\min\{n, 2000\}$ nodes V^* with the highest degrees from V. For every node $v \in V^*$, we generate i_v from $[d]$ uniformly at random and generate r_{v,i_v} from $[0, 0.3]$ uniformly at random. For every $v \in V^*$, we set $S_v = \{i_v\}$ and $q_{v,i_v}(x) = 1-(1-r_{v,i_v})^x$; for every $v \in V \setminus V^*$, $S_v = \emptyset$. This simulates the scenario where marketing efforts are focused on top connected nodes in the network.

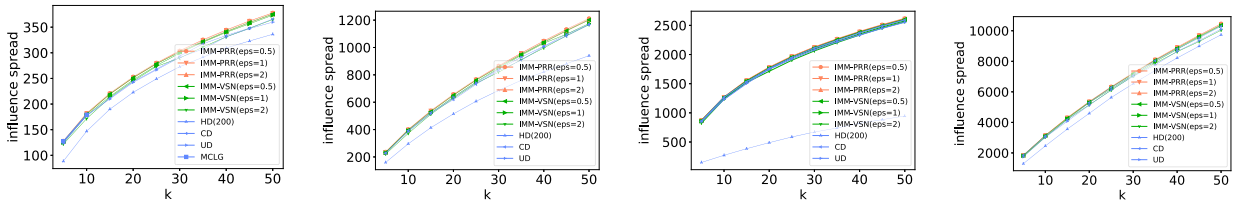
We test both the nonpartitioned budget and partitioned budget cases. For the nonpartitioned cases, we vary the single budget k in our tests, while, for the partitioned cases, we randomly separate the strategies (in both the personalized marketing and the segmented event marketing scenarios) into two roughly equal-sized groups and assign the same budget of $k/2$ to each group, where k is the total budget that we vary in our tests.

3) Algorithms in Comparison: We test the following algorithms.

1) IMM-PRR/IMM-VSN*:* For both algorithms, we set $\ell = 1$, and $\varepsilon = 0.5, 1, 2$. When $\varepsilon = 1$ or 2, IMM-PRR/IMM-VSN no longer has the approximation guarantee, but it is still a valid heuristic algorithm since all other baselines are heuristic algorithms.
2) UD*:* UD is proposed in [16] for personalized marketing. For each discount $c \in \{0.1, 0.2, \ldots, 1\}$, it will return a vector $\mathbf{x}$ s.t. $x_i = 0$ or $x_i = c$ ($i \in [d]$). Then, they run an exhaustive search of c to find a best c.
3) CD*:* CD is also proposed in [16]. CD uses the output of UD as the initial value and runs a coordinate descent algorithm to achieve better result.
4) HD*:* HD is a heuristic baseline, where we choose top M nodes with the highest degrees from V and then distribute the budget to those M nodes proportional to their degrees. We set $M = 200$ in our experiments.
5) MCLG*:* This is L-Greedy (see Algorithm 1) with Monte Carlo simulations to estimate influence spread $g(\mathbf{x})$. We use 100 000 simulations for each estimation of $g(\mathbf{x})$. This number is larger than the typical heuristic setting of 10 000 or 20 000 used in the previous influence maximization studies [1], [6], [8] because, under the same budget k, we may need more greedy iterations (a factor of δ^{-1} in the personalized marketing scenario), and thus, each greedy step needs more accurate evaluation (as suggested by the analysis in [23, Th. 3.6]). We also apply the lazy evaluation method [36] to speed up the greedy process. We remark that this corresponds to the greedy algorithm given in [26] with Monte Carlo simulations for function evaluation.

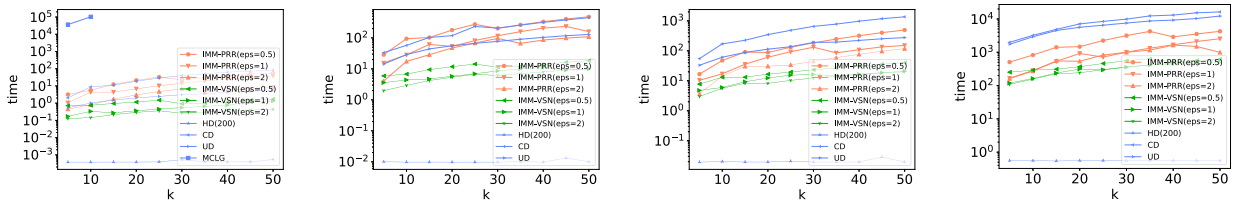
For the personalized marketing scenario, we test all algorithms with granularity $\delta = 0.1$. For the segmented event marketing scenario, we do not test UD, CD, and HD since they are all designed for personalized marketing scenarios. In this case, $\delta = 1$ as required by the scenario. For all cases, we test the total budget k from 5 to 50. We do not include the original influence maximization algorithm IMM for seed set optimization in our tests because [16] already demonstrates that the original IMM is inferior to UD and CD in influence spread.

We remark that for some other algorithms on submodular maximization that we mentioned in the related work, such as the continuous greedy algorithm of [25] and the gradient-based



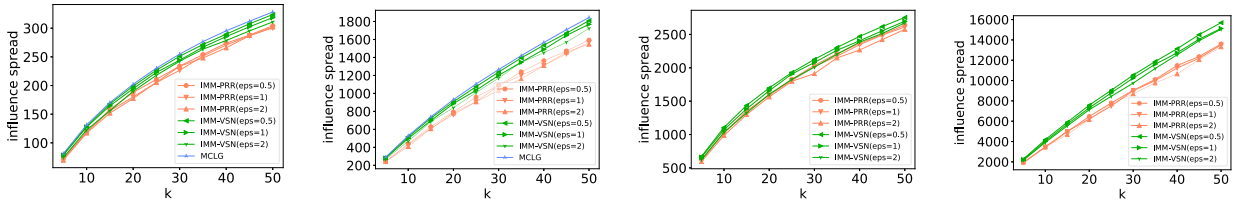
(a) (b) (c) (d)

Fig. 1. Influence spread in personalized marketing scenario. (a) DM. (b) NetHEPT. (c) Flixster. (d) DBLP.



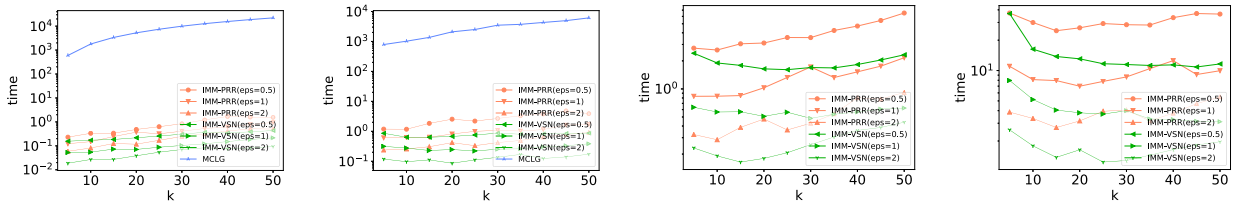
(a) (b) (c) (d)

Fig. 2. Running time in the personalized marketing scenario. (a) DM. (b) NetHEPT. (c) Flixster. (d) DBLP.



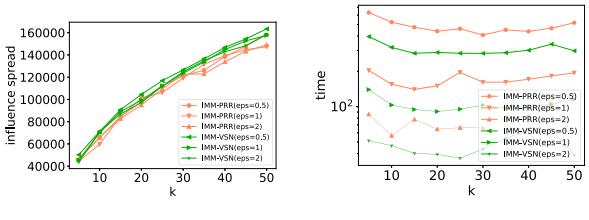
(a) (b) (c) (d)

Fig. 3. Influence spread in the segmented event marketing scenario. (a) DM. (b) NetHEPT. (c) Flixster. (d) DBLP.



(a) (b) (c) (d)

Fig. 4. Running time in the segmented event marketing scenario. (a) DM. (b) NetHEPT. (c) Flixster. (d) DBLP.



(a) (b)

Fig. 5. Influence spread and running time in the segmented event marketing scenario for data set Orkut, nonpartitioned budget case. (a) Orkut: influence spread. (b) Orkut: running time.

algorithms of [28], they are not scalable, and they are for the continuous solution space, and thus, we do not include them in our experiments.

All our tests are run on a Ubuntu 14.04.5 LTS server with 3.3 GHz and 125-GB memory. All algorithms are coded in C++ and compiled by g++. All results on influence spread are the average of 10 000 simulation runs for any given seed set, and all results on running time are the average of five algorithm runs.

B. Experimental Results

Our first group of results are for the nonpartitioned cases (see Figs. 1–5). Fig. 1 shows the influence spread result, and Fig. 2 shows the running time result for the personalized marketing scenario on the first four data sets. First, comparing our two algorithms IMM-PRR and IMM-VSN, they produce about the same influence spread, but IMM-VSN typically runs much faster than IMM-PRR in many cases close to or more than one order of magnitude for the same parameter setting. This demonstrates that the VSN approach indeed runs faster, matching our theoretical analysis. Moreover, changing ε from 0.5 to 2 significantly improves the running time with very slight or no penalty on influence spread.

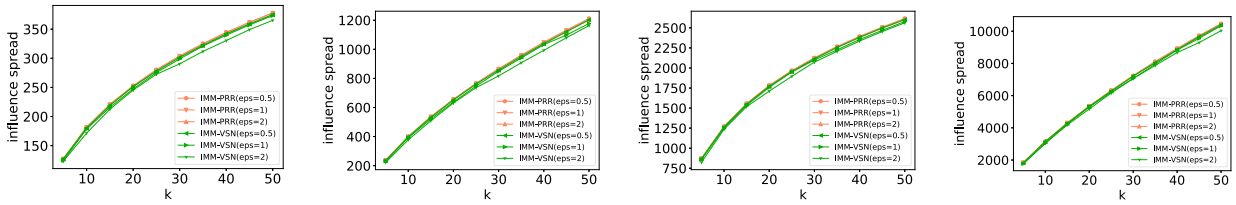


Fig. 6. Influence spread in personalized marketing scenario with partitioned budgets. (a) DM. (b) NetHEPT. (c) Flixster. (d) DBLP.

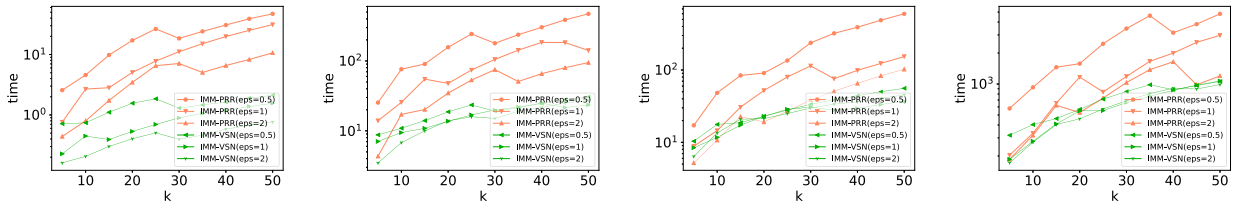


Fig. 7. Running time in the personalized marketing scenario with partitioned budgets. (a) DM. (b) NetHEPT. (c) Flixster. (d) DBLP.

The MCLG algorithm is too slow to run in most cases, so we only run them on the DM data set with $k = 5$ and 10. The result clearly shows that the running times of our algorithms IMM-PRR/IMM-VSN are four to five orders of magnitude faster than MCLG, while their influence spreads are almost the same as MCLG.

When comparing with UD and CD heuristics, our IMM-PRR/IMM-VSN algorithms consistently perform better than UD and CD in influence spread. For running time, IMM-VSN runs much faster than UD and CD by one or two orders of magnitude, and IMM-PRR with $\varepsilon = 2$ is also faster than UD and CD (except on DM). Moreover, when moving from the small data set DM to the large data set DBLP, we see that the running time of UD and CD is getting worse comparing with our algorithms, IMM-PRR/IMM-VSN, indicating that UD/CD algorithms do no scale as well as our algorithms. Overall, the results demonstrate the scalable design of our approach, in particular, our algorithm IMM-VSN with $\varepsilon = 0.5$ can provide both theoretical guarantee and superior empirical performance in both influence spread and running time, while neither UD or CD provides any theoretical guarantee.

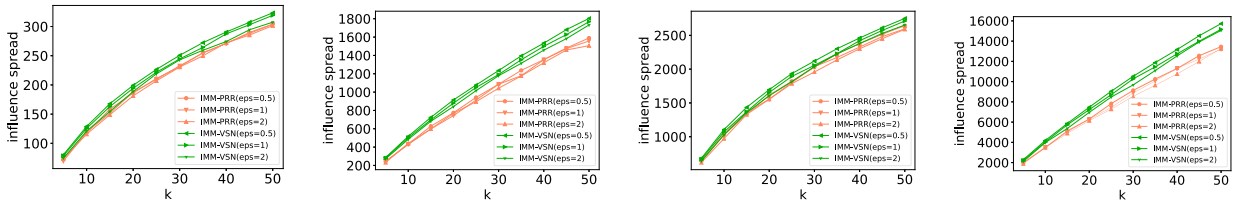
For the baseline heuristic HD, the result shows that its influence spread is significantly lower than others (especially in NetHEPT and Flixster), and thus, it is not a competitive heuristic even though it is very simple and fast.

The results on segmented event marketing are shown in Figs. 3–5. For this test, we also include the largest data set Orkut, as shown in Fig. 5. MCLG is slow, so it is only run on the two small data sets DM and NetHEPT. Overall, the results are consistent with the results of personalized marketing. IMM-VSN typically runs much faster than IMM-PRR, and it runs three to five orders of magnitude faster than MCLG. Increasing ε also significantly improve running time, with only a slight decrease in influence spread. Even when running on the largest data set Orkut with 3M nodes and 117M edges and a much higher average degree than other data sets, our algorithms run reasonable fast, with the fastest IMM-VSN with $\varepsilon = 2$ finishing within 50 s and the slowest IMM-PRR with $\varepsilon = 0.5$ finishing in 700 s. When comparing the running time of the personalized marketing scenario (Fig. 4 versus Fig. 2), we can see that our algorithms run, in general, one to two orders of magnitude faster in the segmented marketing scenario because it is on a much smaller strategy space than the personalized marketing scenario.

In terms of influence spread, IMM-VSN with $\varepsilon = 0.5$ has the best influence spread among different settings for IMM-PRR/IMM-VSN and is only slightly lower than the influence spread achieved by MCLG. The fact that the Monte Carlo greedy algorithm MCLG has the best influence spread is consistent with previous studies on influence maximization [6], [8], [10], which usually shows that the Monte Carlo greedy is among the best in influence spread. In our case, because we use 100 000 Monte Carlo simulations for more accurate function evaluations, while the best accuracy setting for IMM-VSN is $\varepsilon = 0.5$; this makes the best case of IMM-VSN with $\varepsilon = 0.5$ slightly below MCLG in influence spread.

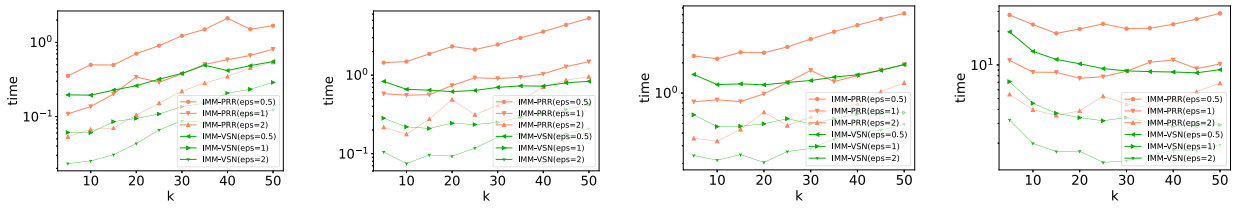
Comparing the influence spread of IMM-PRR with that of IMM-VSN, we see that, in this scenario, IMM-VSN seems to consistently provide slightly better influence spread than IMM-PRR (especially when comparing with the same ε setting). This essentially means that, in this case, with the same number of RR sets, IMM-VSN is able to achieve better influence spread estimates than IMM-PRR. We suspect that this is because of our setting of $q_{v,i_v}(x) = 1 - (1 - r_{v,i_v})^x$ for this scenario compounding with smaller strategy dimensions and a relatively small budget, making that the VSN representations could use a smaller number of virtual nodes to more accurately estimate the influence spread of strategies.

Our second group of results is for the partitioned LIM setting (see Figs. 6–10). As explained in the experimental setup section, for both the personalized marketing and segmented event marketing scenarios, we partition the strategies into two roughly equal-sized groups, and each receives half the budget. For this test, since other baseline algorithms, such as UD, CD, and MCLG, are not designed for this setting, we only run



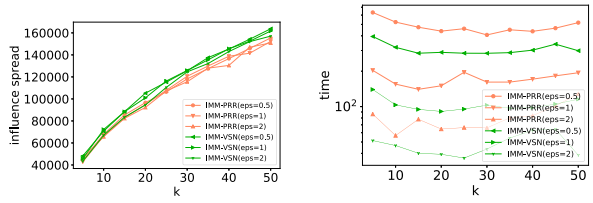
(a) (b) (c) (d)

Fig. 8. Influence spread in the segmented event marketing scenario with partitioned budgets. (a) DM. (b) NetHEPT. (c) Flixster. (d) DBLP.



(a) (b) (c) (d)

Fig. 9. Running time in the segmented event marketing scenario with partitioned budgets. (a) DM. (b) NetHEPT. (c) Flixster. (d) DBLP.



(a) (b)

Fig. 10. (a) Influence spread and (b) running time in the segmented event marketing scenario for data set Orkut, partitioned budget case.

our algorithms with different ε settings. The overall results shown in Figs. 6–10 are consistent with the nonpartitioned case: in general, the influence spread of IMM-PRR and IMM-VSN is close, while IMM-VSN runs faster than IMM-PRR.

From these experiments, we can conclude that for the large class of independent strategy activation scenarios, IMM-VSN is the best choice that provides both theoretical guarantee and fast running time, and it outperforms the Monte Carlo greedy algorithm by several orders of magnitude and is also significantly faster than other competing heuristic algorithms. On the other hand, IMM-PRR works for a more general class of settings beyond independent strategy activation scenarios. Moreover, our algorithms allow the easy tuning of parameter ε to significantly improve running time with small or no penalty on influence spread.

VI. Conclusion and Future Work

We design two RIS-based scalable algorithms: IMM-PRR based on partial RR sets and IMM-VSN based on VSNs that guarantee $1 - 1/e - \varepsilon$ approximation to the LIM problem. IMM-PRR could solve the general LIM problem, while IMM-VSN has better running time for the case of independent strategy activations, as demonstrated both empirically and through theoretical analysis.

Even though IMM-VSN performs better than IMM-PRR in the case of independent strategy activation, in this article, we still present the full algorithm IMM-PRR in detail, including its optimization L-GreedyDelta for independent strategy activation. As we mentioned in Section I and at the beginning of Section III, our rationale is to present a detailed implementation and analysis of these two natural design choices for a fair comparison. We hope that this would be beneficial to our readers to understand the complete details of these design choices and avoid the effort of repeating one approach. Moreover, IMM-PRR is more general than IMM-VSN, covering more scenarios beyond independent strategy activation. For example, if two strategies could have an interaction or complementary effect on a node, causing the node to have a higher chance to be activated than the independent influence of each strategy, then this would be represented by an activation function $h(\mathbf{x})$ that is not the simple aggregation of independent activation attempts, as given in (2). It would be a very interesting research direction to study such more complicated activation functions and its impact on algorithm design—how would the IMM-PRR perform in these settings and can we implement more efficient algorithms than IMM-PRR?

There are also other future directions to this article. One direction is to study continuous domain and investigate if RIS-based approach can be adapted to the continuous domain. Another direction is to study lattice or continuous influence maximization in other influence propagation settings, such as competitive influence maximization.

Appendix

The remaining part of the proof of Theorem 1 is directly modified from the proof of [11, Th. 4] together with the fix in [31].

Lemma 6: Given k, δ, d, n, ℓ, Algorithm 3 returns a $(1-1/e-\varepsilon)$-approximation with at least $1-1/n^{\ell}$ probability if θ, the size of $\mathcal{R}$, is at least $\lambda^*(\ell)/\mathrm{OPT}$, where $\lambda^*(\ell)$ is defined in (6).

Proof: Denote $\mathbf{x}^*$ as the solution of Algorithm 3 and $\mathbf{x}^\circ$ as the optimal solution of LIM problem. Through replacing the

number of possible k-seed set $\binom{n}{k}$ of [11, Lemmas 3 and 4] by the number of possible allocations M in our problem, we can derive that with $1 - n^{\ell}$ probability

$$\hat{g}_{\mathcal{R}}(\mathbf{x}^{\circ}) \geq \left(1 - \varepsilon \cdot \frac{\alpha}{(1-1/e)\cdot\alpha + \beta}\right) \cdot \mathrm{OPT}$$

and

$$\hat{g}_{\mathcal{R}}(\mathbf{x}^{*}) \leq g(\mathbf{x}^{*}) + \left(\varepsilon - \frac{(1-1/e)\varepsilon\alpha}{(1-1/e)\alpha + \beta}\right) \cdot \mathrm{OPT}.$$

Then, by combining the greedy property that $\hat{g}_{\mathcal{R}}(\mathbf{x}^{*}) \geq (1-1/e)\hat{g}_{\mathcal{R}}(\mathbf{x}^{\circ})$, we have $g(\mathbf{x}^{*}) \geq (1-1/e-\varepsilon)\cdot \mathrm{OPT}$. □

Lemma 7: Let ℓ be the input of Algorithm 4. With at least $1 - 1/2n^{(\ell+\gamma)}$ probability, Algorithm 4 returns a set $\mathcal{R}$ of RR sets with $|\mathcal{R}| \geq \lambda^{*}(\ell+\gamma)/\mathrm{OPT}$, where $\lambda^{*}(\ell)$ is defined in (6) and γ is obtained in line 2.

Proof: Through replacing the number of possible k-seed set $\binom{n}{k}$ of [11, Lemmas 6 and 7] by the number of possible allocations M in our problem, we can easily get the result of this lemma. It's $1 - 1/2n^{(\ell+\gamma)}$ rather than $1 - 1/n^{\ell}$ because we reset ℓ as $\ell = \ell + \gamma + \ln 2/\ln n$ in line 3 in Algorithm 4. □

Proof of Theorem 1 (Sketch): By the argument given in [31], when combining Lemmas 6 and 7, we should first take a union bound for $|\mathcal{R}|$ going through $\lambda^{*}(\ell+\gamma)/\mathrm{OPT}$ to $\lambda^{*}(\ell+\gamma)$, and for each fixed length $\mathcal{R}$, we apply Lemma 6 (with ℓ set to $\ell + \gamma + \log 2/\log n$). This would properly show that with probability at most $1/n^{\ell}$, the $\mathcal{R}$ returned by the Sampling procedure will not lead to an output of Algorithm 3 as a $(1-1/e-\varepsilon)$-approximate solution to the LIM problem.

For time complexity, when $q_{v,j}$'s are such that the optimal solution is at least as good as the best single node influence spread, we can have the inequality $\mathrm{EPT} \leq m \cdot \mathrm{OPT}/n$, where EPT is the expected number of incoming edges pointing to nodes in a random RR set [10]. By Lemma 5 and an analysis similar to [11], we can show that the total expected running time is bounded by

$$\begin{aligned} &O\left(k\delta^{-1}\left(\max_{v\in V}|S_v|\right)(\mathrm{EPT}+1)\cdot\frac{\lambda^{*}(\ell+\gamma+\log 2/\log n)}{\mathrm{OPT}}\right) \\ &= O\left(k\delta^{-1}\left(\max_{v\in V}|S_v|\right)\cdot\lambda^{*}(\ell)\cdot(n+m)\right) \\ &= O\left(k\delta^{-1}\left(\max_{v\in V}|S_v|\right)(M+\ell\log n)(n+m)/\varepsilon^{2}\right). \end{aligned}$$

In the second inequality, besides applying $\mathrm{EPT} \leq m\cdot\mathrm{OPT}/n$, we also ignores γ and $\log 2/\log n$ because asymptotically they are all constants.

VII. Remaining Part of the Proof of Theorem 3

We now give the additional details need to prove Theorem 3. The main thing that we want to clarify is the impact that we use a binary search for the reverse sampling in the LT model part from each real node back to each strategy's virtual node. To do so, we need to reformulate a previous result $n\cdot\mathrm{EPT} \leq m\cdot\mathrm{OPT}$ in a more general setting. Let d'_u be the time needed for one-step reverse sampling from node u (previously this would be simply the in-degree of u). Given an RR set R, let $\omega'(R) = \sum_{u\in R} d'_u$. Let $\mathrm{EPT}' = \mathbb{E}[\omega'(R)]$, and EPT' is the expected running time to generate one RR set. Let $\tilde{v}$ be a random real node sampled from V with probability proportional to d'_v's. Then, we have

Lemma 8: $n\cdot\mathrm{EPT}' = \sum_u d'_u \cdot \mathbb{E}[\sigma(\{\tilde{v}\})]$.

Proof (Sketch): The proof essentially follows the proof of [10, Lemma 4], but we need to replace the incoming edges of a node u in that proof to d'_u virtual elements of u so that d'_u matches with the in-degree d_u of u. □

Note that $\sigma(\{\tilde{v}\})$ defined in the abovementioned lemma refers to the classical influence spread of $\tilde{v}$ in the original graph. We are now ready to prove Theorem 3.

Proof of Theorem 3 (Sketch): The approximation ratio is ensured by Theorem 2 and the correctness of the IMM algorithm. For the time complexity, due to our adaption of IMM, the running time is better than the one obtained by simply plugging in the number of nodes $n + k\delta^{-1}d$ and the number of edges $m + k\delta^{-1}\sum_{v\in V}|S_v|$ into the running time formula of IMM. The analysis follows the same structure as that of IMM, and we sketch the main part in the following.

For the greedy NodeSelection procedure, given a sequence of RR sets $\mathcal{R}$ of G_A as input, its running time is $O(\sum_{R\in\mathcal{R}}|R\cap U|)$. The term $|R\cap U|$ is because we only use virtual nodes as seeds, and thus, only the virtual nodes in an RR sets play a role in the NodeSelection algorithm. In fact, we could define an RR set in this case to only contain virtual nodes, but, for the convenience of analyzing the running time, we still keep real nodes in the RR sets. From the analysis in [11] and [31], we know that the total expected running time from all calls to NodeSelection is $O(\mathbb{E}[\theta]\cdot\mathbb{E}[|R\cap U|])$, where θ is the total number of RR sets generated by the algorithm. Similarly, the time spent on generating all RR sets is $O(\mathbb{E}[\theta]\cdot\mathbb{E}[\omega'(R)])$. Since $\omega'(R)$ is the running time of generating R, we have $|R\cap U| \leq \omega'(R)$. Therefore, the total expected running time of the algorithm is $O(\mathbb{E}[\theta]\cdot\mathbb{E}[\omega'(R)]) = O(\mathbb{E}[\theta]\cdot\mathrm{EPT}')$.

By Lemma 8 and the assumption that the optimal solution of the LIM is at least as large as the optimal single node influence spread, we have $\mathrm{EPT}' \leq m'\cdot\mathrm{OPT}/n$. From [11], we know that $\mathbb{E}[\theta] = O(\lambda^{*}/\mathrm{OPT})$. By (6), $\lambda^{*} = O(M + \ell\log n)$. Finally, $m' = \sum_{v\in V} d'_v = O(m + \log(k\delta^{-1})\sum_{v\in V}|S_v|)$ because, for the original graph, the reverse sampling via the triggering set uses time proportional to the in-degree of v in the original graph, and for the virtual nodes, the reverse sampling from each real node to each strategy's virtual nodes takes $O(\log(k\delta^{-1}))$ time via a binary search. Combining all the above together, we know that the expected running time is $O(\mathbb{E}[\theta]\cdot\mathrm{EPT}') = (M+\ell\log n)(m+\log(k\delta^{-1})\sum_{v\in V}|S_v|)/\varepsilon^{2})$. □

References

[1] D. Kempe, J. M. Kleinberg, and E. Tardos, "Maximizing the spread of influence through a social network," *Theory Comput.*, vol. 11, no. 4, pp. 105–147, 2015.

[2] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. M. VanBriesen, and N. S. Glance, "Cost-effective outbreak detection in networks," in *Proc. KDD*, 2007, pp. 420–429.

[3] C. Budak, D. Agrawal, and A. El Abbadi, "Limiting the spread of misinformation in social networks," in *Proc. WWW*, 2011, pp. 665–674.

[4] X. He, G. Song, W. Chen, and Q. Jiang, "Influence blocking maximization in social networks under the competitive linear threshold model," in *Proc. SDM*, 2012, pp. 463–474.

[5] P. Shakarian, J. Salmento, W. R. Pulleyblank, and J. Bertetto, "Reducing gang violence through network influence based targeting of social programs," in *Proc. KDD*, 2014, pp. 1829–1836.
[6] W. Chen, Y. Wang, and S. Yang, "Efficient influence maximization in social networks," in *Proc. KDD*, 2009, pp. 199–208.
[7] A. Goyal, W. Lu, and L. V. S. Lakshmanan, "SIMPATH: An efficient algorithm for influence maximization under the linear threshold model," in *Proc. ICDM*, 2011, pp. 211–220.
[8] C. Wang, W. Chen, and Y. Wang, "Scalable influence maximization for independent cascade model in large-scale social networks," *Data Mining Knowl. Discovery*, vol. 25, no. 3, pp. 545–576, Nov. 2012.
[9] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier, "Maximizing social influence in nearly optimal time," in *Proc. SODA*, 2014, pp. 946–957.
[10] Y. Tang, X. Xiao, and Y. Shi, "Influence maximization: Near-optimal time complexity meets practical efficiency," in *Proc. SIGMOD*, 2014.
[11] Y. Tang, Y. Shi, and X. Xiao, "Influence maximization in near-linear time: A martingale approach," in *Proc. SIGMOD*, 2015, pp. 1539–1554.
[12] E. Cohen, D. Delling, T. Pajor, and R. F. Werneck, "Sketch-based influence maximization and computation: Scaling up with guarantees," in *Proc. CIKM*, 2014, pp. 629–638.
[13] W. Lu, W. Chen, and L. V. Lakshmanan, "From competition to complementarity: Comparative influence diffusion and maximization," *Proc. PVLDB Endowment*, vol. 9, no. 2, pp. 60–71, 2015.
[14] M. Gomez-Rodriguez, L. Song, N. Du, H. Zha, and B. Schölkopf, "Influence estimation and maximization in continuous-time diffusion networks," *ACM Trans. Inf. Syst.*, vol. 34, no. 2, pp. 1–33, Apr. 2016.
[15] S. Chen, J. Fan, G. Li, J. Feng, K.-L. Tan, and J. Tang, "Online topic-aware influence maximization," *Proc. VLDB Endowment*, vol. 8, no. 6, pp. 666–677, Feb. 2015.
[16] Y. Yang, X. Mao, J. Pei, and X. He, "Continuous influence maximization: What discounts should we offer to social network users?" in *Proc. SIGMOD*, 2016, pp. 727–741.
[17] P. Domingos and M. Richardson, "Mining the network value of customers," in *Proc. KDD*, 2001, pp. 57–66.
[18] M. Richardson and P. Domingos, "Mining knowledge-sharing sites for viral marketing," in *Proc. KDD*, 2002, pp. 61–70.
[19] W. Chen, Y. Yuan, and L. Zhang, "Scalable influence maximization in social networks under the linear threshold model," in *Proc. ICDM*, 2010, pp. 88–97.
[20] K. Jung, W. Heo, and W. Chen, "IRIE: Scalable and robust influence maximization in social networks," in *Proc. ICDM*, Dec. 2012, pp. 918–923.
[21] H. T. Nguyen, M. T. Thai, and T. N. Dinh, "Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks," in *Proc. SIGMOD*, 2016, pp. 695–710.
[22] J. Tang, X. Tang, X. Xiao, and J. Yuan, "Online processing algorithms for influence maximization," in *Proc. SIGMOD*, 2018, pp. 991–1005.
[23] W. Chen, L. V. Lakshmanan, and C. Castillo, *Information and Influence Propagation in Social Networks*. San Rafael, CA, USA: Morgan & Claypool, 2013.
[24] E. D. Demaine *et al.*, "How to influence people with partial incentives," in *Proc. WWW*, 2014, pp. 937–948.
[25] M. Feldman, J. Naor, and R. Schwartz, "A unified continuous greedy algorithm for submodular maximization," in *Proc. FOCS*, 2011, pp. 570–579.
[26] T. Soma, N. Kakimura, K. Inaba, and K. Kawarabayashi, "Optimal budget allocation: Theoretical guarantee and efficient algorithm," in *Proc. ICML*, 2014, pp. 351–359.
[27] S. H. Hassani, M. Soltanolkotabi, and A. Karbasi, "Gradient methods for submodular maximization," in *Proc. NIPS*, 2017, pp. 5843–5853.
[28] W. Chen, W. Zhang, and H. Zhao, "Gradient method for continuous influence maximization with budget-saving considerations," in *Proc. AAAI*, 2020.
[29] T. Soma and Y. Yoshida, "A generalization of submodular cover via the diminishing return property on the integer lattice," in *Proc. NIPS*, 2015, pp. 847–855.
[30] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of the approximations for maximizing submodular set functions," *Math. Program.*, vol. 14, no. 1, pp. 265–294, 1978.
[31] W. Chen, "An issue in the martingale analysis of the influence maximization algorithm IMM," in *Proc. CSoNet*, 2019, pp. 286–297.
[32] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey, "An analysis of approximations for maximizing submodular set functions—II," in *Polyhedral Combinatorics* (Mathematical Programming Studies), vol. 8. Berlin, Germany: Springer, 1978, pp. 73–87.
[33] J. Tang, J. Sun, C. Wang, and Z. Yang, "Social influence analysis in large-scale networks," in *Proc. KDD*, 2009, pp. 807–816.
[34] N. Barbieri, F. Bonchi, and G. Manco, "Topic-aware social influence propagation models," in *Proc. IEEE ICDM*, Dec. 2012, pp. 81–90.
[35] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee, "Measurement and analysis of online social networks," in *Proc. 5th ACM/Usenix Internet Meas. Conf. (IMC)*, San Diego, CA, USA, Oct. 2007, pp. 29–42.
[36] M. Minoux, "Accelerated greedy algorithms for maximizing submodular set functions," in *Proc. 8th IFIP Conf. Optim. Techn.*, 1978, pp. 234–243.

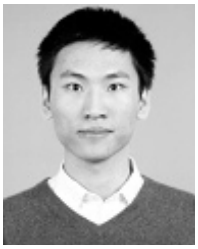
Wei Chen (Fellow, IEEE) received the bachelor's and master's degrees from the Department of Computer Science and Technology, Tsinghua University, Beijing, China, and the Ph.D. degree from the Department of Computer Science, Cornell University, Ithaca, NY, USA.

He is a Principal Researcher at Microsoft Research Asia, Beijing. He is also an Adjunct Professor with the Institute of Interdisciplinary Information Sciences, Tsinghua University, and an Adjunct Researcher with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing. His main research interests include social and information networks, online learning, algorithmic game theory, Internet economics, distributed computing, and fault tolerance.

Dr. Chen is a member of the Technical Committees of Big Data and Theoretical Computer Science of the Chinese Computer Federation.

Ruihan Wu received the B.E. degree in computer science from Tsinghua University, Beijing, China, in 2018. She is currently pursuing the Ph.D. degree with the Computer Science Department, Cornell University, Ithaca, NY, USA.

Her research interests lie in machine learning in general.



Zheng Yu received the B.S. degree in computational mathematics from the University of Science and Technology of China, Hefei, China, in 2017. He is currently pursuing the Ph.D. degree with the Department of Operations Research and Financial Engineering, Princeton University, Princeton, NJ, USA.

His research interests include machine learning theory and stochastic optimization.