# Efficient Approximation Algorithms for Minimum Cost Seed Selection with Probabilistic Coverage Guarantee

CHEN FENG, The Hong Kong Polytechnic University, Hong Kong SAR
XINGGUANG CHEN, National University of Singapore, Singapore
QINTIAN GUO*, The Chinese University of Hong Kong, Hong Kong SAR
FANGYUAN ZHANG, The Chinese University of Hong Kong, Hong Kong SAR
SIBO WANG, The Chinese University of Hong Kong, Hong Kong SAR

Given a social network $G$, a cost associated with each user, and an influence threshold $\eta$, the minimum cost seed selection problem (MCSS) aims to find a set of seeds that minimizes the total cost to reach $\eta$ users. Existing works are mainly devoted to providing an expected coverage guarantee on reaching $\eta$, classified as MCSS-ECG, where their solutions either rely on an impractical influence oracle or cannot attain the expected influence threshold. More importantly, due to the expected coverage guarantee, the actual influence in a campaign may drift from the threshold evidently. Thus, the advertisers would like to request for a probability guarantee of reaching $\eta$. This motivates us to further solve the MCSS problem with a probabilistic coverage guarantee, termed MCSS-PCG.

In this paper, we first propose our algorithm CLEAR to solve MCSS-ECG, which reaches the expected influence threshold without any influence oracle or influence shortfall but a practical approximation ratio. However, the ratio involves an unknown term (i.e., the optimal cost). Thus, we further devise the STAR method to derive a lower bound of the optimal cost and then obtain the first explicit approximation ratio for MCSS-ECG. In MCSS-PCG, it is necessary to estimate the probability that the current seeds reach $\eta$, to decide when to stop seed selection. To achieve this, we design a new technique named MRR, which provides efficient probability estimation with a theoretical guarantee. With MRR in hand, we propose our algorithm SCORE for MCSS-PCG, whose performance guarantee is derived by measuring the gap between MCSS-ECG and MCSS-PCG, and applying the theoretical results in MCSS-ECG. Finally, extensive experiments demonstrate that our algorithms achieve up to two orders of magnitude speed-up compared to alternatives while meeting the requirement of MCSS-PCG with the smallest cost.



CCS Concepts: • **Theory of computation → Graph algorithms analysis**.

Additional Key Words and Phrases: Minimum Cost Seed Selection; Approximation Algorithms; Probability Estimation

**ACM Reference Format:**
Chen Feng, Xingguang Chen, Qintian Guo, Fangyuan Zhang, and Sibo Wang. 2024. Efficient Approximation Algorithms for Minimum Cost Seed Selection with Probabilistic Coverage Guarantee. *Proc. ACM Manag. Data* 2, N4 (SIGMOD), Article 197 (September 2024), 26 pages. https://doi.org/10.1145/3677133

*Qintian Guo is the corresponding author.
This work was conducted while Chen Feng and Xingguang Chen were at CUHK.

Authors' addresses: Chen Feng, chenfeng@polyu.edu.hk, The Hong Kong Polytechnic University, Hong Kong SAR; Xingguang Chen, xgchen@nus.edu.sg, National University of Singapore, Singapore; Qintian Guo, qtguo@link.cuhk.edu.hk, The Chinese University of Hong Kong, Hong Kong SAR; Fangyuan Zhang, fzhang@se.cuhk.edu.hk, The Chinese University of Hong Kong, Hong Kong SAR; Sibo Wang, swang@se.cuhk.edu.hk, The Chinese University of Hong Kong, Hong Kong SAR.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.
2836-6573/2024/9-ART197 $15.00
https://doi.org/10.1145/3677133

## 1 INTRODUCTION

Social networks are becoming increasingly prevalent for people to share their thoughts, ideas, and opinions etc. Based on the social relationships among individuals, a piece of information could quickly go viral via the "word-of-mouth" effect. With this effect, in social advertising, advertisers could recruit only a subset of users (called *seeds*) as the sources to trigger a large cascade of influence in the network. As the diffusion proceeds, the number of users engaged in the campaign may reach some *tipping point*, from which the idea, trend, or behavior would spread like wildfire [15, 32, 41]. Thus, the advertisers would like to influence a target number of users to reap greater benefits. Such demands give rise to the MCSS (i.e., Minimum Cost Seed Selection) problem, which aims to find a set of seeds that minimizes the cost to achieve an influence threshold $\eta$. Existing works on MCSS mainly focus on reaching the threshold in expectation [18, 20, 22, 27, 31, 47, 52], categorized as MCSS-ECG (i.e., MCSS with Expected Coverage Guarantee). However, actually these works cannot produce a seed set that reaches the influence threshold with an approximation guarantee in practice. For examples, some works [31, 52] require an influence oracle to select the seeds, which is not available in real campaigns, while other works [18, 20, 22, 27, 47] suffer from a shortfall in reaching the influence threshold.

More importantly, since the attainment of $\eta$ is guaranteed in *expectation*, even if MCSS-ECG is well settled, the selected seeds can only guarantee the coverage when the campaign is carried out for plenty of times, while this is often not the case in practice. For example, in 2020, Dell launched 15 influencer marketing campaigns on 8 social platforms (e.g., YouTube, Instagram, and Snapchat) to promote its computers to the global market, finally reaching over 168 million users in total [16]. Even if the 15 campaigns were performed on the same platform, the deviation between the actual influence and the expected influence is as large as 0.4 times of the expectation with a probability of 99%, by a simple calculation from the concentration bound. Alternatively, the advertisers would request for a probability guarantee $p$ of the actual influence reaching the threshold [26, 38], obtaining direct information on the likelihood of campaign success. To meet this requirement, Zhang et al. take the head in solving the MCSS problem with a probability guarantee on the influence spread, termed MCSS-PCG [50].

Although substantial contributions have been made by Zhang et al., their work cannot adapt to the practice well. First, the estimation error on their influence spread is required to be a very small value, inversely proportional to $\eta$ times the network scale, which is not realistic in practice. Second, numerous Monte Carlo simulations are carried out after each seed selection to examine whether the probability guarantee is satisfied, which are quite time-consuming. Finally, it is assumed that the costs to engage different users into the campaign are the same, which however should be different. In fact, it is widely reported that the cost is closely related to the number of followers of an influencer. For example, on Instagram, typically a nano-influencer with 1k to 10k followers costs \$10–100, while a mega-influencer with at least 1 million followers usually charges over \$10,000 for one post [8]. These shortcomings hinder the solution of [50] from being applied to real-world commercial campaigns.

In this paper, we strive to design efficient approximation algorithms for the MCSS-PCG problem under non-uniform user costs. Looking into MCSS-PCG, we find that the problem is NP-hard and non-submodular. Thus, it is hard to directly solve it with a theoretical guarantee. Nevertheless, as one may see, the seed set that solves MCSS-PCG will also achieve some expected influence spread $\tilde{\eta}$. Assume we select seeds for MCSS-PCG and MCSS-ECG with exactly the same criteria. Let the upper bound (resp. lower bound) of $\tilde{\eta}$ be $\tilde{\eta}^{\top}$ (reps. $\tilde{\eta}^{\perp}$). Then, by bounding the cost to reach an expected influence of $\tilde{\eta}^{\top}$ from $\tilde{\eta}^{\perp}$, and referring to the theoretical guarantee in MCSS-ECG to reach $\tilde{\eta}^{\perp}$, we can derive an approximation ratio of solving MCSS-PCG. However, as aforementioned,

MCSS-ECG has not been well solved yet. Thus, we first retrospect MCSS-ECG and devise the algorithm CLEAR to solve it without any influence shortfall or influence oracle. After massive analyses, we derive the approximation ratio of CLEAR, which however is found to be implicit, a function of the optimal cost. Thus, we further come up with a method STAR to provide an explicit approximation ratio for CLEAR.

To solve the MCSS-PCG problem, we have to address another major challenge, i.e., efficient and accurate estimation of the probability that the current influence reaches $\eta$. Note that the idea of Monte Carlo estimation is to collect numerous diffusion samples from the seeds [50]. To be efficient in sampling, we attempt to borrow the idea of reverse sampling from the reverse reachable set (RR-set) [4, 45], which is generated by initializing a stochastic BFS from a random user along the reversed edge directions. However, naive application of RR-sets would result in evident estimation bias, even if necessary modification is made. We find the main cause lies in the independency among RR-sets. Specifically, when the RR-sets are independent, the states of an edge in different reverse diffusion processes that produce the RR-sets may conflict with each other, and thus could not form an overall diffusion sample. Thus, we require the reverse diffusion to be consistent with each other, by prohibiting other reverse diffusion processes from revising the state of an edge once it has been traversed. Accordingly, we develop the MRR technique, which provides accurate probability estimation theoretically. Based on the above efforts, we propose our algorithm SCORE to solve MCSS-PCG with a performance guarantee.

To summarize, we highlight our main contributions as follows.

- In this paper, we first revisit the MCSS-ECG problem to solve it without any influence shortfall or influence oracle. On this basis, we make the initial attempt to solve the MCSS-PCG problem efficiently under the more practical non-uniform costs setting.
- Specifically, we propose the CLEAR algorithm to tackle MCSS-ECG, which ensures the attainment of $\eta$ with an approximation ratio related to the optimal cost. Since the optimal cost is unknown, we invent the STAR method to derive its lower bound and obtain the first explicit approximation ratio for an algorithm reaching $\eta$.
- Next, we devise the MRR technique to estimate the probability that the influence of a seed set could reach $\eta$, accompanied with a theoretical guarantee for the estimation accuracy. Actually, MRR is not restricted to the settings of MCSS, and thus provides a general tool for problems in need of influence probability estimation.
- Equipped with the MRR, we design our algorithm SCORE for MCSS-PCG. Analytically, we first bound the cost to solve MCSS-PCG based on MCSS-ECG. Then, by referring to the ratio in MCSS-ECG, we derive the performance guarantee of SCORE.
- Finally, extensive experiments are conducted to evaluate our proposed algorithms. The results show that our SCORE algorithm meets the requirement of MCSS-PCG with the smallest cost and up to two orders of magnitude speed-up, compared with the state-of-the-art. Moreover, our CLEAR algorithm could always reach the influence threshold with the least running time in MCSS-ECG.

In Table 1, we further compare CLEAR and SCORE to the state-of-the-art algorithms to show our advantages over them, and the detailed description can be found in Appendix B of [1].

## 2 PRELIMINARIES

### 2.1 Problem Formulation

The social network is modeled as a directed graph $G = (V, E)$, where $V$ is the set of users and $E$ is the set of relationships with $|V| = n$ and $|E| = m$. Each user $u \in V$ is associated with some cost $c(u) \in \mathbb{R}^+$, and each edge $e = (u, v) \in E$ is associated with some probability $p(e) \in [0, 1]$, called the propagation probability. Especially, the cost of a set of users $S$ is defined as $c(S) = \sum_{u \in S} c(u)$. For

Table 1. Comparison with existing algorithms.

| E/PCG | Algorithms | Feasibility | Different costs | $\epsilon$ | Approximation ratio |
|---|---|---|---|---|---|
| ECG | BCGC [20] | $(1-\alpha)\eta$ | ✓ | $\leq \frac{\alpha}{3}$ | $\approx 1+\ln\frac{3-\alpha}{\alpha}$ |
| | TEGC [20] | $(1-\alpha)\eta$ | ✓ | $\leq \frac{\alpha}{3}$ | $\approx 1+\ln\frac{3-\alpha}{\alpha}$ |
| | Our CLEAR | ✓ | ✓ | ✓ | $c(S)/c^{\perp}(S^o)$ |
| PCG | MinSeed [50] | ✓ | × | $\leq o(\frac{1}{n\eta})$ | $\lceil\frac{\ln 2\eta n}{n-\eta}\rceil c(S^o)+\frac{(c+c')n}{n-\eta-c'}+3$ |
| | ASTI [43] | ✓ | × | ✓ | $\frac{(1+\ln\eta)^2}{(1-1/e)(1-\epsilon)}$ |
| | Our SCORE | ✓ | ✓ | ✓ | $\frac{c(S_I)}{c^{\perp}(S^*)}c(S^o)+\frac{(a+a')n\rho}{n-\eta-a'}+2$ |

each edge $e=(u,v)$, user $u$ is said to be the in-neighbor of $v$ and user $v$ is said to be the out-neighbor of $u$. Further, the in-neighbors of user $u$ are denoted as the set $N_{\text{in}}(u)$. Moreover, we say $e$ is the out-edge of $u$ and the in-edge of $v$.

To depict the diffusion process from seeds, we adopt the widely applied *independent cascade (IC) model* for modeling, while our results could be easily extended to the other popular model named *linear threshold (LT) model*. Specifically, in the IC model, given an arbitrary seed set $S$, the influence diffusion expands in discrete steps as follows. Initially, at time step 0, only users in $S$ are activated while other users remain inactivated. Then, at each time step $i>0$, each user (e.g., $u$) that is newly activated in the previous step has a *single* chance to activate its out-neighbors (e.g., $v$) via edge $e=(u,v)$. Accordingly, user $v$ becomes activated with probability $p(e)$. This process proceeds until no user could be further activated. Note that the activation attempts of users are independent of each other and each activated user would not turn back to be inactivated. Finally, the number of users activated, a random variable w.r.t. the propagation process, is called the influence spread of $S$, denoted as $I(S)$, where $I(\cdot)$ is the influence function $I: 2^V \to \mathbb{R}^+ \cup \{0\}$. Then, the expected influence spread could be written as $\mathbb{I}(S)=\mathbb{E}[I(S)]$. On this basis, the number of users required to be influenced, in expectation $\mathbb{I}(\cdot)$ or in random $I(\cdot)$, is denoted as $\eta$. Further, we define the truncated influence with regard to $\eta$ as $\Gamma_\eta(S)=\min\{\mathbb{I}(S),\eta\}$, where the subscript $\eta$ is omitted when the context is clear.

Lemma 2.1 indicates that the expected influence $\mathbb{I}(\cdot)$ satisfies both monotonicity and submodularity. Then, the truncated influence $\Gamma(S)$ is also monotone and submodular, since the two properties would not be affected by truncation.

Lemma 2.1 ([25]). *The expected influence function $\mathbb{I}(\cdot)$ is both monotone and submodular. That is, given an arbitrary set of users $T\subseteq V$,*

$$\forall S\subseteq T, \text{ we have } \mathbb{I}(S)\leq\mathbb{I}(T);$$
$$\forall S\subseteq T, \forall u\in V\backslash T, \text{ we have } \mathbb{I}(S\cup\{u\})-\mathbb{I}(S)\geq\mathbb{I}(T\cup\{u\})-\mathbb{I}(T).$$

Next, we introduce an equivalent view of the diffusion process [25]. Before the diffusion starts, the states of edges are claimed to be unknown, since none of them is probed. As the diffusion proceeds, the edges are gradually probed by active users and the diffusion through an edge is revealed to be either successful or not. Accordingly, the states of edges are declared to be *live* or *blocked*, and encoded as "1" or "0" respectively. To realize the diffusion through an edge $(u,v)$, we could flip a coin with bias $p(u,v)$. From this view, it is clear that whether the coin is flipped at the moment $u$ is activated or at the very beginning of diffusion would not affect the outcome. Thus, we could flip a coin for each edge, store the result, and just reveal it when the in-neighbor is activated. On this basis, we present the concept of *realization* in Definition 2.2.

*Definition 2.2 (Realization).* Given the network $G=(V,E)$ and the probability $p(e)$ of each edge $e\in E$, a realization $\phi$ is a function that maps each edge to its state, i.e., $\phi: E\to\{0,1\}$.

Now we are ready to define the MCSS problems in Definition 2.3, where both MCSS-ECG and MCSS-PCG are formalized. Note that the main difference between the two problems lies in their constraints on the seed set $S$. MCSS-ECG asks the *expected* influence spread $\mathbb{I}(S)$ to reach $\eta$, while MCSS-PCG requires the probability of a *random* influence spread $I(S)$ reaching $\eta$ to be at least $p$.

*Definition 2.3.* Given the social network $G = (V, E)$, costs of users $c(u)$ $(\forall u \in V)$, propagation probabilities $p(e)$ $(\forall e \in E)$, the probability threshold $p$, and the influence threshold $\eta$,
(1) **MCSS-ECG** aims to find a set of users $S^o \subseteq V$ that could activate at least $\eta$ users in expectation with the minimum cost. Formally,

$$S^o = \arg\min_{S:\mathbb{I}(S)\geq\eta} c(S).$$

(2) **MCSS-PCG** aims to find a set of users $S^o \subseteq V$ that activates at least $\eta$ users with probability at least $p$ with the minimum cost, i.e.,

$$S^o = \arg\min_{S:\mathbb{P}[I(S)\geq\eta]\geq p} c(S).$$

By reduction from the uniform cost case in [50] and the set cover problem [11], we can establish Proposition 2.4, which shows that the two MCSS problems in Definition 2.3 are both NP-hard, and are hard to be approximated within a ratio smaller than $\ln n$.

Proposition 2.4. *Both MCSS-ECG and MCSS-PCG problems are NP-hard, and for any $\epsilon > 0$, MCSS problem cannot be approximated within a ratio of $(1-\epsilon)\ln n$ unless NP has $n^{O(\log\log n)}$-time deterministic algorithms.*

## 2.2 Existing Solutions

**MCSS-ECG.** Early works on MCSS-ECG are only theoretically tractable or practically efficient. Notably, Han et al. attempt to solve the problem with both theoretical guarantees and practical efficiency [20]. To this end, they apply RR-sets in Definition 2.5 for efficient and analysable influence estimation [4] [45]. Specifically, a random RR-set can be generated in two steps. First, we select a user from $V$ as the *root* uniformly at random. Second, we initialize an influence propagation from the root, along the reverse directions of edges, while the diffusion probability of each edge is still the same as the original IC model. Finally, the users that are activated during the reverse diffusion process comprise a random RR-set. Given a collection of RR-sets, the expected influence could be estimated with a provable error. On this basis, they first propose their basic algorithm BCGC. Then, TEGC is devised to generate RR-sets in a "trial-and-error" manner, aiming to terminate RR-set generation in advance.

However, to derive the theoretical guarantee for their algorithms, Han et al. have to set aside a shortfall $\alpha$ by only reaching an expected influence of $(1-\alpha)\eta$, where the shortfall $\alpha$ must be non-zero to ensure their approximation ratio (i.e., $1 + \ln\frac{3-\alpha}{\alpha}$) is bounded. Thus, BCGC and TEGC actually could not reach the influence threshold $\eta$ with any performance guarantee. The other drawback lies in the number of RR-sets they need. In their analysis, the collection of RR-sets is required to guarantee an estimation error of $\gamma$ for all possible seed sets that fail to reach $(1-\alpha)\eta$, while the number of such seed sets is exponential w.r.t. $\eta$. And, to this end, the union bound is applied to restrict the probability of estimation failure. Then, the resultant RR-sets number is derived to be $\Theta(n\ln\frac{n}{\eta}/\gamma^2)$, where $\gamma$ is required to be smaller than $\alpha$. To be close to $\eta$, the shortfall $\alpha$ is often set to be a small value, and so is $\gamma$, resulting in huge computational cost.

*Definition 2.5 (RR-set [4]).* Given a realization $\phi$ of the network $G$ and a root user $v$, an RR-set for $v$ is the set of users that could reach $v$ in $G$ under $\phi$. Accordingly, a random RR-set is an RR-set generated from a random root under a random realization.

**MCSS-PCG.** The MCSS-PCG problem is first explored by Zhang et al. in their work [50]. To derive the performance guarantee, the authors first look into MCSS-ECG and derive the approximation ratio of the *seed number* to reach the influence threshold $\eta$, by properly setting a shortfall and compensating it. Then, the authors study the MCSS-PCG problem and propose the MinSeed algorithm to reach the influence threshold with a probability at least $p$. Especially, after selecting a seed, the probability of current influence reaching $\eta$ is estimated by the Monte Carlo method. Then, based on the analysis in MCSS-ECG, the authors derive an approximation ratio of MinSeed and bound it with the variance of influence spread.

The work of Zhang et al. provides the first solution to MCSS-PCG with a performance guarantee. Significant advance as it is, their algorithm MinSeed is not that practical, since they require the error of influence estimation to be smaller than $\frac{n-\eta}{8n^2(\eta+1)} = \Omega(\frac{1}{8n\eta})$, which is quite stringent and hard to be achieved in practice. Moreover, the Monte Carlo method is applied to estimate the probability of reaching $\eta$ after selecting each seed. By Hoeffding's inequality, hundreds of simulations have to be performed each time, even for a moderate estimation error. By our experiments, to select enough seeds, typically more than $10^5$ simulations have to be conducted in total, which would take large amounts of time. Further, the costs of users are assumed to be uniform. Then, the total cost becomes simply the number of seed users, facilitating their analysis. However, this setting makes their theoretical results deviate from real-world scenarios, where the costs to invite different users are actually different, as reported in the Introduction.

**Adaptive Seed Minimization (ASM).** Tang et al. have made significant contributions to ASM in their work [43], where seed users are iteratively selected based on the observation of the diffusion from previous seeds. Due to the adaptive nature, the seed selection would not stop until the influence spread is observed to reach $\eta$. Thus, the influence threshold will always be reached in ASM, and the seed set of ASM is certainly a feasible solution to both MCSS-PCG and MCSS-ECG.

To conform with the theoretical results in [17], Tang et al. argue that in the adaptive setting, the truncated influence should be considered, instead of the original influence function. Note that their truncated influence is defined w.r.t. each realization, i.e., $\Gamma(S) = \sum_{\phi} \min\{I_{\phi}(S), \eta\}$, where $I_{\phi}(S)$ is the influence of $S$ under realization $\phi$. To estimate the truncated influence, Tang et al. propose the multi-root reverse reachable (mRR) sets. Specifically, each mRR-set is generated by initializing a reverse propagation from $k$ roots selected from $V$ uniformly at random. The set of users activated by the $k$ roots together form an mRR-set $R$. Accordingly, the mRR-set $R$ is said to be covered by $S$, if $S \cap R \neq \emptyset$. Then, the truncated influence of any seed set $S$ is estimated as its coverage times $n$ and divided by the number of mRR-sets, which is not an unbiased estimation but the error is bounded. On this basis, Tang et al. develop the ASTI algorithm that achieves an approximation ratio of $\frac{(\ln \eta+1)^2}{(1-1/e)(1-\epsilon)}$, where $\epsilon \in (0, 1)$ is the estimation error.

ASTI is often shown to reach $\eta$ with fewer seed users. This advantage is mainly due to its adaptive setting, which allows sensible seed selection based on previous diffusion results. However, it is also the adaptivity that results in its inefficiency. To explain, the number of roots in an mRR-set is required to be the quotient between $n - I(S_i)$ and $\eta - I(S_i)$ in [43]. After selecting a new seed, the quotient will change due to the increased $I(S_i)$. Thus, after each round of seed selection, all mRR-sets must be regenerated from scratch, which takes much computation cost. Moreover, they only consider the uniform cost setting, while in practice, users often demand different costs. It seems that ASTI could adapt to non-uniform costs by selecting users with the maximum influence-to-cost ratio. However, under this criterion, users with small costs yet moderate influence are more likely to be selected, leading to an increased number of seed selection rounds and more regeneration of new mRR-sets. Moreover, to ensure the estimation accuracy of small influence, more mRR-sets are needed in each round, aggravating the scalability issue.

### 2.3 Notations and Useful Tools

Before presenting our solutions, we would like to introduce some notations frequently used in subsequent sections. To begin with, the seed set output by our algorithms is denoted as $S = \bigcup_{j=1}^{k}\{s_i\}$, where $s_i$ is the $i$-th seed selected and $k$ is the number of seeds. Especially, the seed set $S_i = \bigcup_{j=1}^{i}\{s_i\}$ is composed of the first $i$ seeds and $S_0$ is defined as the empty set. The optimal seed set is denoted as $S^o = \{s_1^o, s_2^o, \cdots, s_h^o\}$. Further, the maximum (resp. minimum) cost of users is written as $c_{\max} = \max\{c_i\}$ (resp. $c_{\min} = \min\{c_i\}$) and their quotient is denoted as $\rho = c_{\max}/c_{\min}$.

Lemma 2.6. *[45] Given a collection of reverse reachable sets $R_1, R_2, \cdots, R_\theta$ and the seed set $S$, for any $\epsilon \geq 0$, we have*

$$\mathbb{P}[\Lambda(S) \geq (1+\epsilon)\mathbb{I}(S)] \leq \exp\left(-\frac{\epsilon^2\theta\mathbb{I}(S)}{2+2\epsilon/3}\right), \tag{1}$$

$$\mathbb{P}[\Lambda(S) \leq (1-\epsilon)\mathbb{I}(S)] \leq \exp\left(-\frac{\epsilon^2\theta\mathbb{I}(S)}{2}\right). \tag{2}$$

For influence estimation, we also apply the RR-sets method in [45, 46]. Given a collection of RR-sets $\mathcal{R}$, an RR-set $R \in \mathcal{R}$ is said to be *covered* by a seed set $S$ if $S \cap R \neq \emptyset$, and the *coverage* $\Lambda(S)$ of the seed set $S$ is the number of RR-sets covered by $S$. Then, we could estimate the expected influence of $S$ by $\hat{\mathbb{I}}(S) = n\Lambda(S)/\theta$, where $\theta$ is the number of RR-sets in $\mathcal{R}$. It is easy to verify that $\hat{\mathbb{I}}(S)$ is an unbiased estimation of $\mathbb{I}(S)$. Moreover, the estimation error could be bounded by Lemma 2.6 [45]. For ease of notation, we further denote $\eta_i = \eta - \Gamma(S_i)$ and its estimation $\hat{\eta}_i = (1+\epsilon)\eta - \hat{\Gamma}(S_i)$, where $\hat{\Gamma}(S_i) = \min\{\hat{\mathbb{I}}(S_i), (1+\epsilon)\eta\}$ and $\epsilon$ is the estimation error of $\hat{\mathbb{I}}(S_i)$. For more notations used in this paper, please refer to Appendix A of [1].

## 3 MCSS-ECG

In this section, we solve the MCSS-ECG problem without the requirement of influence shortfall or influence oracle. To this end, we first present the CLEAR algorithm, whose approximation ratio is revealed to be a function of the optimal cost $c(S^o)$. Since $c(S^o)$ is unknown, we further devise the method STAR to derive the lower bound of $c(S^o)$, and then an explicit approximation ratio of CLEAR is available by replacing $c(S^o)$ with its lower bound. Due to space limitations, we omit the proofs of all the theoretical results, while interested readers may refer to Appendix C in [1].

### 3.1 The CLEAR Algorithm

Specifically, we propose the CLEAR algorithm to solve MCSS-ECG, the details of which are shown in Alg. 1. Different from previous works on MCSS-ECG, our algorithm CLEAR involves two collections of RR-sets, $\mathcal{R}_1$ for seed selection, and $\mathcal{R}_2$ for influence verification. The new RR-set collection $\mathcal{R}_2$ is introduced to verify whether the current seed set could reach the influence threshold $\eta$.

After initializing the seed set $S$, $\mathcal{R}_1$, and $\mathcal{R}_2$, we determine the number of RR-sets needed by $\mathcal{R}_1$ and $\mathcal{R}_2$ with Lemma 3.1. To elaborate, we require the failure probabilities of $\mathcal{R}_1$ and $\mathcal{R}_2$ to be both $\delta$, while the estimation error of $\mathcal{R}_1$ is $\epsilon_e$ and the verification error of $\mathcal{R}_2$ is $\epsilon_v$. To guarantee the verification quality, $\epsilon_v$ is often set to be a smaller value. However, the size of $\mathcal{R}_2$ (i.e., $\theta_v$) is still often much smaller than that of $\mathcal{R}_1$ (i.e., $\theta_e$), since the minimum influence whose estimation accuracy has to be guaranteed by $\mathcal{R}_1$ and $\mathcal{R}_2$, is different. To explain, $\mathcal{R}_1$ is applied to select the seeds based on their influence. Thus, $\mathcal{R}_1$ should guarantee the estimation accuracy of seeds' influence, whose minimum value is the influence of the first seed, denoted as $\xi$. While $\mathcal{R}_2$ only needs to guarantee the estimation accuracy of the final influence $\eta$. Since $\eta$ is much larger than $\xi$, we often have $\theta_e \gg \theta_v$. Thus, the generation of $\mathcal{R}_2$ would not incur much computation cost.

**Algorithm 1:** CLEAR

**Input:** Graph $G = (V, E)$, user costs $c(\cdot)$, diffusion probability $p(\cdot)$, failure probability $\delta$, estimation error $\epsilon_e$, minimum influence $\xi$, verification error $\epsilon_v$.

**Output:** The seed set $S$.

$S \leftarrow \emptyset, \mathcal{R}_1 \leftarrow \emptyset, \mathcal{R}_2 \leftarrow \emptyset$;

$\theta_e \leftarrow \lceil (2 + \frac{2\epsilon_e}{3}) \ln \frac{2}{\delta} n \big/ (\epsilon_e^2 \xi) \rceil$;

$\theta_v \leftarrow \lceil (2 + \frac{2\epsilon_v}{3}) \ln \frac{2}{\delta} n \big/ (\epsilon_v^2 \eta) \rceil$;

**while** $|\mathcal{R}_1| < \theta_e$ **do**

  Generate a random RR-set and insert it into $\mathcal{R}_1$;

**while** $|\mathcal{R}_2| < \theta_v$ **do**

  Generate a random RR-set and insert it into $\mathcal{R}_2$;

**while** $\hat{\Gamma}_2(S) < (1 + \epsilon_v)\eta$ **do**

  $u^* \leftarrow \max \frac{\hat{\Gamma}_1(S \cup \{u\}) - \hat{\Gamma}_1(S)}{c(u)}$;

  $S \leftarrow S \cup \{u^*\}$;

**return** $S$;

Lemma 3.1. *To estimate $\mathbb{I}(S)$ ($\mathbb{I}(S) \geq \xi$) with estimation error $\epsilon$ and failure probability $\delta$, the number of RR-sets should satisfy*

$$\theta \geq (2 + \frac{2\epsilon}{3}) \ln \frac{2}{\delta} n \Big/ \epsilon^2 \xi. \tag{3}$$

Given $\mathcal{R}_1$ and $\mathcal{R}_2$, we are ready to select the seeds iteratively. In each round $i$, we select the user $u^*$ with the largest marginal influence-to-cost ratio estimated by $\mathcal{R}_1$ as the seed $s_i$, forming seed set $S_i$. The seed selection will not terminate until the influence estimated by $\mathcal{R}_2$, i.e., $\hat{\Gamma}_2(S)$, is at least $(1 + \epsilon_v)\eta$. Since the validation error of $\mathcal{R}_2$ is at most $\epsilon_v$, the expected influence of the final seed set $S$ would be at least $\eta$, satisfying the requirement of MCSS-ECG.

Next, we present theoretical analyses on our algorithm CLEAR. To begin with, we first measure the estimation accuracy of $\hat{\Gamma}(S)$ and $\hat{\eta}_i$ by Lemmas 3.2 and 3.3 respectively.

Lemma 3.2. *If $(1-\epsilon)\mathbb{I}(S) \leq \hat{\mathbb{I}}(S) \leq (1+\epsilon)\mathbb{I}(S)$, we have the same estimation error for the truncated influence, i.e.,*

$$(1 - \epsilon)\Gamma(S) \leq \hat{\Gamma}(S) \leq (1 + \epsilon)\Gamma(S). \tag{4}$$

Lemma 3.3. *If $(1 - \epsilon)\Gamma(S_i) \leq \hat{\Gamma}(S_i) \leq (1 + \epsilon)\Gamma(S_i)$, $\epsilon \in (0, \frac{1}{2})$, and $\epsilon' = \frac{2\epsilon}{1-2\epsilon}$, then the estimation accuracy of $\hat{\eta}_i$ satisfies*

$$(1 - \epsilon)\left[\hat{\eta}_i - \epsilon' \hat{\Gamma}(S_i)\right] \leq \eta_i \leq (1 + \epsilon)\hat{\eta}_i. \tag{5}$$

On this basis, we could derive the approximation ratio of CLEAR by Theorem 3.4. Further, in Proposition 3.5 we justify the rationale of the approximation ratio obtained in Theorem 3.4.

Theorem 3.4. *Let $\beta = \min_i \frac{\hat{\Gamma}_1(S_i) - \hat{\Gamma}_1(S_{i-1})}{c(s_i)}$. When $\epsilon = o(\frac{1}{\eta \rho^2})$, the total cost $c(S)$ to reach the influence threshold $\eta$ by Alg. 1 satisfies*

$$c(S) \leq c(S^o)\left[1 + \frac{1}{B_\epsilon} \ln \frac{(1+\epsilon)\eta}{\beta c(S^o) - k C_\epsilon}\right], \tag{6}$$

*where $B_\epsilon = \frac{(1+\epsilon')(1-\epsilon)}{1+\epsilon-2\rho\epsilon+\rho\epsilon'(1-\epsilon)}$ and $C_\epsilon = \frac{2\epsilon'(1-\epsilon^2)\eta\rho}{1+\epsilon-2\rho\epsilon+\rho\epsilon'(1-\epsilon)}$.*

**Algorithm 2:** STAR

**Input:** RR-set collections $\mathcal{R}_1, \mathcal{R}_2$ with $|\mathcal{R}_1|=\theta_e$, $|\mathcal{R}_2|=\theta_v$.
**Output:** The seed set $S$ and approx. ratio $\alpha$.

$S \leftarrow \emptyset, c^{\perp}(S^o) = c_{\min}, c(S) = 0$;
**while** $\hat{\Gamma}_2(S) < (1+\epsilon_v)\eta$ **do**
  $u^* \leftarrow \max \frac{\hat{\Gamma}_1(S\cup\{u\})-\hat{\Gamma}_1(S)}{c(u)}$;
  $S \leftarrow S \cup \{u^*\}$;
  Calculate $\epsilon_i$ with $\mathcal{R}_1$ by Lemma 3.6;
  Calculate $c_i^{\perp}(S^o)$ with $\mathcal{R}_2$ and $\epsilon_i$ by Lemma 3.7;
  $c^{\perp}(S^o) \leftarrow \max\{c^{\perp}(S^o), c_i^{\perp}(S^o)\}$;
  $c(S) \leftarrow c(S) + c(u^*)$;
$\alpha = c(S)/c^{\perp}(S^o)$;
**return** $S, \alpha$;

Proposition 3.5. *The approximation ratio* $\left(1 + \frac{1}{B_\epsilon} \ln \frac{(1+\epsilon)\eta}{\beta c(S^o) - kC_\epsilon}\right)$ *in Theorem 3.4 is well defined, since it always holds that*

$$0 < \beta c(S^o) - kC_\epsilon < (1+\epsilon)\eta.$$

Recall that the estimation error required by [50] in their Theorem 2 is $\frac{n-\eta}{8n^2(\eta+1)}$. Under the same setting of uniform costs, where $\rho = c_{\max}/c_{\min} = 1$, our restriction $\epsilon = o(\frac{1}{\eta})$ is much looser and thus easier to be achieved, which is at least $n$ times larger than that required by [50]. Further, as can be seen, our algorithm could reach the influence threshold without allowing a shortfall $\alpha$ like [20]. Moreover, note that the number of RR-sets we need, dominated by $\theta_e$, is also much smaller than that in [20], which is of order $\Theta(n \ln \frac{n}{\eta}/\gamma^2)$, where $\gamma$ is a constant smaller than $\alpha$.

### 3.2 Practical Guarantee of CLEAR

Although we have established Theorem 3.4 to demonstrate the performance guarantee of CLEAR, the actual approximation ratio remains implicit due to the inclusion of the unknown term $c(S^o)$. To address this issue, we further propose a new method called STAR to derive the lower bound of $c(S^o)$, and thus obtain an explicit approximation ratio for CLEAR.

The details of STAR are summarized in Alg. 2. To carry out the algorithm, we only need the two RR-set collections $\mathcal{R}_1$ and $\mathcal{R}_2$ in CLEAR. Then, a seed set with an explicit approximation ratio could be generated. Thus, STAR could be used to substitute the Lines 8-10 in CLEAR for seed selection.

To begin with, we initialize the seed set with the empty set, the optimal cost $c(S^o)$ with $c_{\min}$, and the cost of the selected seeds with 0. Then, we select seed users with the greedy idea, the same as CLEAR. Specifically, in each round, we select the seed with the maximum influence-to-cost ratio estimated by $\mathcal{R}_1$. Recall Lemma 3.1 that the estimation error of $\mathcal{R}_1$ is related to the influence $\mathbb{I}(S_i)$ under estimation, which is increasing with seed selection. By transforming the concentration bound in Lemma 2.6, we could further derive the estimation error $\epsilon_i$ of $\mathcal{R}_1$ in each round $i$ in Lemma 3.6.

Lemma 3.6. *Given the RR-set collection* $\mathcal{R}_1$*, the seed set* $S_i$*, and the estimator* $\hat{\mathbb{I}}(S) = n\Lambda_1(S_i)/\theta_e$*. Let* $w = \ln(1/\delta)$*. If* $\Lambda_1(S_i) > 2w/3$*, then with probability at least* $1-2\delta$ *the estimation error* $\epsilon_i$ *is upper bounded by*

$$\frac{\Lambda_1(S_i)}{\left(\sqrt{\Lambda_1(S_i) + \frac{2w}{9}} - \sqrt{\frac{w}{2}}\right)^2 - \frac{w}{18}} - 1.$$

Moreover, in each round, after selecting the seed $u^*$, we can immediately derive a lower bound $c_i^{\perp}(S^o)$ of the optimal cost $c(S^o)$ by Lemma 3.7, based on the influence of the current seed set $S_i$. To tighten the lower bound, we further take the largest lower bound among all rounds as the final bound $c^{\perp}(S^o)$. Meanwhile, we record the cost of the selected seeds $c(S)$. Finally, the approximation ratio of CLEAR can be explicitly calculated as $c(S)/c^{\perp}(S^o)$, due to $c(S)/c(S^o) \leq c(S)/c^{\perp}(S^o)$. Note that, the ratio is enlarged in the inequality, since we are approximating a minimization problem.

Lemma 3.7. *After selecting the i-th seed ($i > 0$), we could derive a lower bound of $c(S^o)$ by*

$$c(S^o) \geq c_i^{\perp}(S^o) = \frac{c(s_i)\left[\eta - (1+\epsilon_i)\hat{\Gamma}(S_{i-1})\right]}{(1+\epsilon_i)\left[\hat{\Gamma}(S_{i-1}\cup\{s_i\}) - \hat{\Gamma}(S_{i-1})\right] + 2\epsilon_i\rho\hat{\Gamma}(S_{i-1})}.$$

The proof of Lemma 3.7 is similar to the derivation of the approximation ratio in Theorem 3.4. However, to obtain a tighter bound, we deduce the lower bound of $c(S^o)$ directly based on the influence function $\hat{\Gamma}(S_i)$ instead of $\hat{\eta}_i$.

With Lemma 3.7, we can also make the ratio in Theorem 3.4 explicit by replacing the $c(S^o)$ in the square bracket of Ineq. (6) with its lower bound. Accordingly, the ratio in Ineq. (6) becomes clear and will still hold as long as $\beta c^{\perp}(S^o) > kC_{\epsilon}$.

Moreover, note that the derivation of $c^{\perp}(S^o)$ does not require the number of RR-sets in $\mathcal{R}_1$ to be any specific value. Actually, given any $\mathcal{R}_1$ with $|\mathcal{R}_1| > \frac{2}{3}\ln\frac{1}{\delta}$, we can derive an approximation ratio by STAR. Thus, our algorithm is also applicable to the online setting where the advertiser could pause the generation of $\mathcal{R}_1$ at any time, ask for a seed set and its performance guarantee, and continue to generate RR-sets to improve the quality.

## 4 MCSS-PCG

To solve the MCSS-PCG problem, we have to estimate the probability of $S$ reaching $\eta$. Previous works tend to adopt the Monte Carlo method, which stimulates a number of diffusion samples from the seeds to provide an estimation. In this way, large numbers of samples have to be generated during seed selection, failing to be scalable to large networks. To be efficient in sampling, we attempt to adapt the well-known reverse sampling technique RR-sets for estimation. The main obstacle in applying RR-sets for probability estimation is that each RR-set could only provide an estimation of 0 or 1. No specific probability could be derived. To overcome this issue, we may generalize the generation of an RR-set by selecting more roots to make a more fine-grained estimation. Intuitively, the ratio of the roots covered by $S$ could be viewed as a probability. To elaborate, given a set of $r$ roots, independent RR-sets are generated from each root. Then, the $r$ RR-sets comprise a group of RR-sets $\{R_1, R_2, \cdots, R_r\}$, referred to as GRR. Each RR-set (e.g., $R_i$) in this group makes its binary estimation $\hat{p}_i \in \{0, 1\}$ according to whether the seed set $S$ intersects with it. Then, the average value $\frac{1}{r}\sum \hat{p}_i$ could be regarded as the probability estimated by this group of RR-sets. Finally, numerous such RR-set groups could be generated to make the estimation more precise. This method seems to be promising in both efficiency and accuracy. However, the resultant estimation may deviate from the ground truth evidently. For a vivid explanation, we present an example in Appendix D of [1].

### 4.1 Meta RR-sets

Although RR-set groups are biased, their idea of transforming a binary estimation to a fine-grained decimal estimation still offers valuable insights into probability estimation.

Inspecting the RR-set groups, we find that the cause of their bias may come from the independency among the RR-sets, which is a basic property of the reverse sampling technique though. To explain, since the RR-sets are generated independently, the reverse diffusion processes that generate the RR-sets may belong to different realizations. As a result, the states of an edge in different RR-sets may

conflict with each other. Thus, a group of RR-sets could not comprise an overall diffusion sample like a Monte Carlo simulation. To address this issue, we propose a new reverse sampling method, called Meta RR-set (MRR-set). Each MRR-set consists of a number of RR-sets that are required to be consistent with the same realization, where the concept of being consistent is presented in Definition 4.1. Especially, each of such RR-sets is called a consistent RR-set (cRR-set), to differentiate from traditional RR-sets. Then, an MRR-set could be viewed as a sample of an overall diffusion process, equivalent to stimulating a diffusion. Thus intuitively, with enough MRR-sets, we could estimate the probability of current influence reaching any given threshold.

*Definition 4.1.* A collection of cRR-sets is *consistent* with the same realization, if the states of an edge are the same in the reverse diffusion processes that generate the cRR-sets.

Given the social network $G$ and diffusion probabilities $p(\cdot)$, we can generate a collection of MRR-sets $\mathcal{M}$ by Alg. 3. Let $\kappa$ be the number of MRR-sets to be generated, and $\theta$ be the number of cRR-sets in an MRR-set. After initializing the MRR-set collection $\mathcal{M}$, we will insert $\kappa$ MRR-sets into $\mathcal{M}$. To generate an MRR-set $\mathcal{M}_i$, we have to adopt two arrays to record the states of users and edges during the reverse propagation (Lines 3-4). The first array $state[\cdot]$ stores the state of each edge $e \in E$, which is shared by all the cRR-sets in $\mathcal{M}_i$, ensuring the cRR-sets are consistent with the same realization. Each element in $state[\cdot]$ may take three values. Initially, the states of all edges are set to be `Unknown`, since no edge has been probed yet. After an MRR-set generation begins, edges will be probed by the users. Then, the state of an edge will become `True` if the activation through it is successful, and `False` otherwise. For each MRR-set, the second array $roots[\cdot]$ stores $\theta$ roots sampled from the network uniformly at random, from which the cRR-sets will be generated.

When generating a cRR-set $cRR_{i,j}$ in $\mathcal{M}_i$, we need another two arrays to keep track of the diffusion process. First, a queue $Q$ is applied to record the set of users whose in-edges have not been probed. Second, to avoid inserting duplicate users into the same cRR-set, the array $added[\cdot]$ is introduced to indicate whether a user has been added into $cRR_{i,j}$ or not. Specifically, an element in $added[\cdot]$ takes value `True` if the corresponding user has already been inserted into the cRR-set, and vice versa.

To generate a cRR-set $cRR_{i,j}$, we take out a root $r$ from the root array $roots$ of $\mathcal{M}_i$ to trigger the reverse diffusion. Initially, the cRR-set $cRR_{i,j}$ and the queue $Q$ only contain the root $r$. Accordingly, all elements in array $added[\cdot]$ is set to be `False` except $added[r]$. Then, for each user $u$ in $Q$, we try to activate its in-neighbors $N_{\text{in}}(u)$. Specifically, for each in-neighbor $v$, we have to first examine the state of the edge $(v, u)$, in case its state has been revealed by other roots. In this way, we could ensure that the edge states are consistent among different cRR-sets. Accordingly, the state of $(v, u)$ (i.e., $state[(v, u)]$) may result in three cases.

(1) If the state of $(v, u)$ is `False` (Line 12), this means that the edge $(v, u)$ has been probed and revealed to be blocked. To keep the edge states consistent with the same realization, we only need to adopt the previous probing result. Thus, user $v$ would remain to be inactivated and no further action is needed.

(2) On the other hand, if the state of $(v, u)$ is found to be `True` (Line 14), then the diffusion through $(v, u)$ will also be successful as before and user $v$ will be activated. Before adding $v$ into $cRR_{i,j}$, we have to further check whether $v$ has already been inserted into $cRR_{i,j}$ in case of duplication, by looking up the array $added[v]$. If $added[v]$ is `False`, we could safely insert $v$ into the cRR-set. Accordingly, we update the state of $added[v]$ by `True` and append $v$ to $Q$ to further probe its in-neighbors. Otherwise, if $added[v]$ is `True`, we just skip user $v$ for the next one.

(3) Different from the above two cases, if the state of the edge $(v, u)$ is still `Unknown` (Line 22), meaning the edge $(v, u)$ has not been probed yet, then we ascertain it by simulating the diffusion through $(v, u)$. Specifically, we sample a number in $[0, 1]$ uniformly at random. If the random

**Algorithm 3:** MRR$(G, p, \kappa, \theta)$

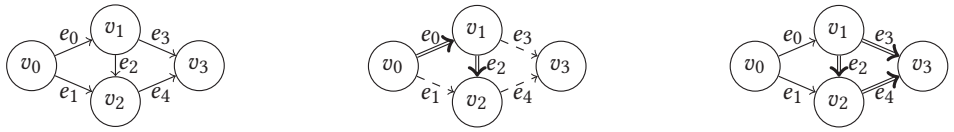
**Input:** Graph $G = (V, E)$, diffusion probability $p(\cdot)$, the number of roots $\theta$, the number of MRR-sets $\kappa$.
**Output:** A collection of MRR-sets $\mathcal{M}$.

1 $\mathcal{M} \leftarrow \emptyset$;
2 **for** $i \in [1, \kappa]$ **do**
3 $\quad \mathcal{M}_i \leftarrow \emptyset$, $state[e] \leftarrow$ Unknown, $\forall e \in E$ ;
4 $\quad roots \leftarrow$ UniformSampling$(V, \theta)$;
5 $\quad$ **for** $j \in [1, \theta]$ **do**
6 $\quad\quad r \leftarrow roots[j]$, $cRR_{i,j}$.append$(r)$;
7 $\quad\quad Q$.clear().append$(r)$;
8 $\quad\quad added[u] \leftarrow$ False, $\forall u \in V$; $added[r] \leftarrow$ True;
9 $\quad\quad$ **while** $Q.empty()$ == False **do**
10 $\quad\quad\quad u \leftarrow Q$.pop();
11 $\quad\quad\quad$ **for** $v \in N_{in}(u)$ **do**
12 $\quad\quad\quad\quad$ **if** $state[(v, u)]$ == False **then**
13 $\quad\quad\quad\quad\quad$ **continue**;
14 $\quad\quad\quad\quad$ **else if** $state[(v, u)]$ == True **then**
15 $\quad\quad\quad\quad\quad$ **if** $added[u]$ == True **then**
16 $\quad\quad\quad\quad\quad\quad$ **continue**;
17 $\quad\quad\quad\quad\quad$ **else**
18 $\quad\quad\quad\quad\quad\quad$ $cRR_{i,j}$.append$(v)$ ;
19 $\quad\quad\quad\quad\quad\quad$ $added[u] \leftarrow$ True;
20 $\quad\quad\quad\quad\quad\quad$ $Q$.append$(v)$ ;
21 $\quad\quad\quad\quad\quad\quad$ **continue** ;
22 $\quad\quad\quad\quad$ **else**
23 $\quad\quad\quad\quad\quad$ **if** $rand() > p(v, u)$ **then**
24 $\quad\quad\quad\quad\quad\quad$ $state[(v, u)] \leftarrow$ False;
25 $\quad\quad\quad\quad\quad$ **else**
26 $\quad\quad\quad\quad\quad\quad$ $state[(v, u)] \leftarrow$ True;
27 $\quad\quad\quad\quad\quad\quad$ **go to** Line 15;
28 $\quad\quad\quad \mathcal{M}_i$.append$(cRR_{i,j})$;
29 $\quad \mathcal{M}$.append$(\mathcal{M}_i)$;
30 **return** $\mathcal{M}$;

number is larger than the diffusion probability $p(v, u)$, then the diffusion fails and the state of $(v, u)$ is set to be False. Otherwise, the edge should be live and its state is set to be True. Accordingly, we try to insert the user $v$ into the cRR-set, by adopting the same actions in Lines 15-21.

When $Q$ becomes empty, we finish the generation of $cRR_{i,j}$ from root $r$, and insert the cRR-set $cRR_{i,j}$ into $\mathcal{M}_i$. Then, iterating the above processes for the $\theta$ roots, we can derive the MRR-set $\mathcal{M}_i$. After generating $\kappa$ such MRR-sets, we obtain the final MRR-set collection $\mathcal{M}$.

*Example 4.2.* To be intuitive, we further illustrate the execution of Alg. 3 with Fig. 1. For ease of reverse diffusion, we first reverse all edge directions to derive the transpose graph $G^T(V, E)$, which consists of four users $(v_0, v_1, v_2, v_3)$, and five edges $(e_0 = (v_0, v_1), e_1 = (v_0, v_2), e_2 = (v_1, v_2), e_3 = (v_1, v_3), e_4 = (v_2, v_3))$ which are associated with the same diffusion probability 0.5. Let us generate




$G^T(V, E)$

$cRR_{1,1} = \{v_0, v_1, v_2\}$
$cRR_{1,2} = \{v_1, v_2\}$

$cRR_{2,1} = \{v_1, v_3, v_2\}$
$cRR_{2,2} = \{v_2, v_3\}$

(a) Transpose graph (b) MRR-set $\mathcal{M}_1$ (c) MRR-set $\mathcal{M}_2$

Fig. 1. Example of MRR-sets generation.

an MRR-set collection with $\kappa = \theta = 2$. To derive the first MRR-set $\mathcal{M}_1$, we assume the array $roots = \{v_0, v_1\}$. When reversely diffusing from root $v_0$, the edges $e_0$ and $e_2$ are probed to be live while the rest are not, as shown in Fig. 1(b). Then, the cRR-set from $v_0$ is $cRR_{1,1} = \{v_0, v_1, v_2\}$, and $state = \{1, 0, 1, 0, 0\}$. For the second root $v_1$, when examining its in-neighbor $v_2$, since the state of $e_2 = (v_1, v_2)$ (i.e., $state[e_2]$) has been revealed to be `True` by the first cRR-set, we directly insert $v_2$ into the $cRR_{1,2}$. However, when probing the in-neighbor $v_3$ of $v_1$, the state of $e_3 = (v_1, v_3)$ has been revealed to be `False` by $cRR_{1,1}$. Thus, user $v_3$ could not be activated or inserted. Similarly, when $v_3$ is probed by $v_2$, it is not inserted either, since the edge $e_4 = (v_2, v_3)$ was also revealed to be `False` by $cRR_{1,1}$. Then, the second cRR-set is $cRR_{1,2} = \{v_1, v_2\}$, and the MRR-set is $\mathcal{M}_1 = \{cRR_{1,1}, cRR_{1,2}\}$.

To generate the second MRR-set $\mathcal{M}_2$, assume the roots are $\{v_1, v_2\}$. For the root $v_1$, when probing the edge $e_3 = (v_1, v_3)$, we assume the diffusion is successful, as shown in Fig. 1(c). Then, user $v_3$ is inserted to the cRR-set, and the array $added = \{0, 1, 0, 1\}$. Likewise, the other in-neighbor $v_2$ is activated and inserted via $e_2 = (v_1, v_2)$. Then, we have to probe the edge $e_4$ from $v_2$, which is found to be live. However, the value of $v_3$ in $added$ is `True`. Thus, user $v_3$ would not be inserted into $cRR_{2,1}$. Accordingly, the cRR-set from $v_1$ is $cRR_{2,1} = \{v_1, v_3, v_2\}$, and the array $state = \{?, ?, 1, 1, 1\}$. For the second root $v_2$, when probing its only in-neighbor $v_3$, we find that the state of the edge $e_4 = (v_2, v_3)$ (i.e., $state[e_4]$) has been revealed to be live by $cRR_{2,1}$, and $added[v_3]$ is `False` in the new cRR-set. Thus, we directly insert $v_3$ into $cRR_{2,2}$. Finally, the cRR-set is $cRR_{2,2} = \{v_2, v_3\}$, and the MRR-set is $\mathcal{M}_2 = \{cRR_{2,1}, cRR_{2,2}\}$.

**Remark.** Besides the probability $\mathbb{P}[I(S) \geq \eta]$, MRR-sets can also be applied to estimate the expected influence $\mathbb{I}[S]$ in traditional influence maximization (IM) problems [21, 23, 48]. To elaborate, note that each MRR-set $\mathcal{M}_i$ is a sketch of a realization $\phi_i$. Then, $\mathcal{M}_i$ can estimate the influence spread under $\phi$ as $\hat{I}_{\phi_i}(S) = n \cdot \Lambda_i(S)/\theta$, where $\Lambda_i(S)$ is the number of cRR-sets covered by $S$ in $\mathcal{M}_i$. With a number of MRR-sets $\{\mathcal{M}_i\}_{i=1}^{\kappa}$, the expected influence $\mathbb{I}(S)$ can be estimated as $\hat{\mathbb{I}}(S) = \frac{1}{\kappa}\sum_{i=1}^{\kappa} \hat{I}_{\phi_i}(S)$. It is easy to see that $\hat{\mathbb{I}}(S)$ is an unbiased estimator and MRR-sets have a smaller variance than traditional RR-sets, where detailed analyses could be found in Appendix C of [1]. However, an MRR-set takes more time to be generated than an RR-set, since each MRR-set has to generate numerous cRR-sets. How to improve the efficiency of MRR-sets to solve traditional IM could be a potential direction for future work.

### 4.2 Theoretical Analysis of MRR-sets

In this subsection, we show that MRR-sets could estimate the probability of $I(S)$ reaching $\eta$ with a theoretical guarantee, by properly setting parameters $\kappa$ and $\theta$ in Alg. 3.

Let the random variable $Y(\Psi, \Phi) = 1$ denote the event that a random user $\Psi$ is activated by the seed set $S$ in a random diffusion process $\Phi$, and $Y(\Psi, \Phi) = 0$ otherwise. Note that the randomness of $Y(\Psi, \Phi)$ comes from two sources: the selection of a random user and the random propagation

from $S$ to the user. Thus, even if we take an expectation w.r.t. user selection, the result $\mathbb{E}_\Psi[Y(\Psi,\Phi)]$ is still a random variable.

To estimate $\mathbb{P}[I(S) \geq \eta]$, an attractive way is to enquire the *complementary* cumulative distribution function (CCDF) $F^c(\eta) = \mathbb{P}[I(S) \geq \eta]$ of $I(S)$ directly, which however is often unavailable. Nevertheless, we find that Lemma 4.3 could connect the distribution of $I(S)$ with $\mathbb{E}_\Psi[Y(\Psi,\Phi)]$, and thus provide the possibility to access $F^c(\cdot)$ in an indirect way.

Lemma 4.3. *The distribution of $n \cdot \mathbb{E}_\Psi[Y(\Psi,\Phi)]$ is the same as $I(S)$, and so is the CCDF of $n \cdot \mathbb{E}_\Psi[Y(\Psi,\Phi)]$.*

Then, we show that actually $\mathbb{E}_\Psi[Y(\Psi,\Phi)]$ could be well approximated by the MRR-sets, based on which we could obtain a surrogate CCDF of $I(S)$. To see this, let us first consider the simpler case where the realization is fixed, i.e., $\Phi = \phi$. In this case, the r.v. $\mathbb{E}_\Psi[Y(\Psi,\phi)]$ is actually a constant value, since whether a user could be activated by $S$ under a given realization $\phi$ is deterministic, by the definition of $\phi$. Next, by Lemma 4.4, we show that a random cRR-set *cRR* of an MRR-set under $\phi$ is actually an unbiased estimator of $\mathbb{E}_\Psi[Y(\Psi,\phi)]$.

Lemma 4.4. *Given any realization $\phi$ and a random cRR-set cRR generated under $\phi$, let the indicator r.v. $Y = 1$ if cRR is covered by $S$ and $Y = 0$ otherwise. Then, we have*

$$\mathbb{E}[Y] = \mathbb{E}_\Psi[Y(\Psi,\phi)].$$

Thus, we can estimate $\mathbb{E}_\Psi[Y(\Psi,\phi)]$ with a number of cRR-sets in an MRR-set. To elaborate, assume we have generated $\theta$ cRR-sets under the realization $\phi$, which form an MRR-set $\mathcal{M}_i$. Let $Y_{i,j}$ be the indicator random variable of the cRR-set $cRR_{i,j}$, where $Y_{i,j} = 1$ if $cRR_{i,j}$ is covered by $S$, and $Y_{i,j} = 0$ otherwise. Given the $\theta$ cRR-sets, we can estimate $\mathbb{E}_\Psi[Y(\Psi,\phi)]$ with the sample mean $\hat{Y}_i = \frac{1}{\theta}\sum_{j=1}^{\theta} Y_{i,j}$. Further, by establishing the concentration results in Lemma 4.5, we can derive the estimation error of $\mathcal{M}_i$. Specifically, Lemma 4.5 implies that, to estimate $\mathbb{E}_\Psi[Y(\Psi,\phi)]$ with an estimation error of $\tau$ and failure probability $\delta_1$, the number of cRR-sets we need in an MRR-set should be at least $\theta = \ln\frac{1}{\delta_1}/(2\tau^2)$.

Lemma 4.5. *Given any realization $\phi$ and an MRR-set $\mathcal{M}_i$ with $\theta$ cRR-sets generated under $\phi$, for any $\tau > 0$ and $\delta_1 = e^{-2\theta\tau^2}$, we have*

$$\mathbb{P}\left[\mathbb{E}_\Psi[Y(\Psi,\phi)] \geq \frac{1}{\theta}\sum_{j=1}^{\theta} Y_{i,j} - \tau\right] \geq 1 - \delta_1. \tag{7}$$

With tractable estimation of $\mathbb{E}_\Psi[Y(\Psi,\phi)]$ under any realization $\phi$, we are ready to consider the original case where the realization is random. To this end, we first introduce some useful concepts and tools. Let $Z_1, Z_2, \cdots Z_\kappa$ be a set of real-valued samples from the same distribution, whose CCDF is denoted as $F^c(z)$ and CDF is $F(z) = 1 - F^c(z)$. Then, we define the empirical CCDF $\hat{F}^c_\kappa(z)$ in Definition 4.6. Moreover, the DKW inequality in Lemma 4.7 harnesses the error between $\hat{F}_\kappa(z)$ and $F(z)$, and thus also harnesses the error between $\hat{F}^c_\kappa(z)$ and $F^c(z)$.

*Definition 4.6.* Given a series of samples $Z_1, Z_2, \cdots Z_\kappa$ from the same distribution whose CCDF is $F^c(z)$, then the empirical CCDF and empirical CDF are defined as

$$\hat{F}^c_\kappa(z) = \frac{1}{\kappa}\sum\nolimits_{i=1}^{\kappa} \mathbb{I}_{\{Z_i \geq z\}}, \quad \hat{F}_\kappa(z) = 1 - \hat{F}^c_\kappa(z),$$

where $\mathbb{I}_{\{Z_i \geq z\}}$ is the indicator function which takes value 1 if $Z_i \geq z$, and 0 otherwise.

Lemma 4.7 (DKW Inequality [33]). *For any integer $\kappa$ and real number $\lambda \geq \min\left\{\sqrt{\ln 2/2\kappa}, \frac{1.0841}{\kappa^{2/3}}\right\}$, it holds for any $z \in \mathbb{R}$ that*

$$\mathbb{P}\left[F(z) \geq \hat{F}_\kappa(z) + \lambda\right] \leq e^{-2\kappa\lambda^2}.$$

**Algorithm 4:** SCORE

**Input:** Graph $G = (V, E)$, user cost $c(\cdot)$, diffusion probability $p(\cdot)$, failure probabilities $\delta, \delta_1$ and $\delta_2$, estimation errors $\epsilon_e, \tau$ and $\lambda$, minimum influence $\xi$.
**Output:** The seed set $S$.

$S \leftarrow \emptyset, \mathcal{R} \leftarrow \emptyset, \mathcal{M} \leftarrow \emptyset, \theta_e \leftarrow \lceil (2 + \frac{2\epsilon_e}{3}) \ln \frac{2}{\delta} n / (\epsilon_e^2 \xi) \rceil$;
**while** $|\mathcal{R}| < \theta_e$ **do**
  Generate a random RR-set and insert it into $\mathcal{R}$;
$\theta \leftarrow \ln \frac{1}{\delta_1} / (2\tau^2), \kappa \leftarrow \ln \frac{1}{\delta_2} / (2\lambda^2)$;
$\mathcal{M} \leftarrow \mathtt{MRR}(G, p, \kappa, \theta)$;
$cnt \leftarrow 0, coverage[i] \leftarrow 0, \forall i \in [1, \kappa]$;
$covered[i][j] \leftarrow \mathtt{False}, \forall i \in [1, \kappa], \forall j \in [1, \theta]$;
**while** $cnt/\kappa < p + \lambda$ **do**
  $u^* \leftarrow \max \frac{\hat{\Gamma}(S \cup \{u\}) - \hat{\Gamma}(S)}{c(u)}$;
  $S \leftarrow S \cup \{u^*\}$;
  $\mathtt{VEST}(u^*, cnt, covered, coverage, \mathcal{M}, \eta/n + \tau)$;
**return** $S$;

The DKW inequality indicates that by sampling a sufficient number of realizations, we could derive an empirical CCDF of $\mathbb{E}_\Psi[Y(\Psi, \Phi)]$ with provable deviation. Note that each MRR-set $\mathcal{M}_i$ corresponds to a realization $\phi_i$. Thus, by generating a series of MRR-sets $\mathcal{M}_1, \mathcal{M}_2, \cdots, \mathcal{M}_\kappa$, we could derive an empirical CCDF $\hat{F}_\kappa^c(\eta) = \frac{1}{\kappa} \sum_{i=1}^{\kappa} \mathbb{I}_{\{n\mathbb{E}_\Psi[Y(\Psi,\phi_i)] \geq \eta\}}$. Recall that $\mathbb{E}_\Psi[Y(\Psi, \phi_i)]$ could be estimated by the cRR-sets in $\mathcal{M}_i$. Thus, we could replace it with $\hat{Y}_i = \frac{1}{\theta} \sum_{j=1}^{\theta} Y_{i,j}$. To ensure the estimated value would not overestimate the probability, we could further restrict the indicator function to take value 1 only when $\mathbb{E}_\Psi[Y(\Psi, \phi_i)]$ reaches $\eta$ with certainty, by incorporating the estimation error $\tau$. Thus, we could define a stricter empirical CCDF as

$$\hat{F}_\kappa^{c'}(\eta) = \frac{1}{\kappa} \sum_{i=1}^{\kappa} \mathbb{I}_{\left\{\frac{1}{\theta} \sum_{j=1}^{\theta} Y_{i,j} \geq \eta/n + \tau\right\}}. \tag{8}$$

On this basis, Theorem 4.8 presents the estimation error $\lambda$ of $\kappa$ MRR-sets. Then, by setting $\kappa = \ln \frac{1}{\delta_2} / (2\lambda^2)$, when $\hat{F}_\kappa^{c'}(\eta) \geq p + \lambda$, we have $\mathbb{P}(I(S) \geq \eta) \geq p$ holds with probability at least $1 - \delta_1 - \delta_2$.

Theorem 4.8. *Given the seed set $S$ and $\kappa$ MRR-sets $\mathcal{M}_1, \mathcal{M}_2, \cdots, \mathcal{M}_\kappa$, where each MRR-set contains $\theta$ cRR-sets, if the empirical CCDF satisfies $\hat{F}_\kappa^{c'}(\eta) \geq p + \lambda$ where $\lambda \geq \min\left\{\sqrt{\ln 2/2\kappa}, 1.0841/\kappa^{2/3}\right\}$, then we have $\mathbb{P}(I(S) \geq \eta) \geq p$ holds with probability at least $1 - e^{-2\kappa\lambda^2} - \kappa e^{-2\theta\tau^2} + \kappa e^{-2\kappa\lambda^2 - 2\theta\tau^2}$.*

### 4.3 The SCORE Algorithm

Although MRR-sets provide reliable probability estimation, naive application of them could not fully explore their potential in efficiency. To explain, when estimating the probability, we have to examine the state of each MRR-set and each cRR-set therein, by verifying whether it is covered by $S$. If we simply check the state of MRR-sets and cRR-sets from scratch after each seed selection, extensive computation would be incurred. Instead, we devise an accumulative estimation method called VEST to apply MRR-sets efficiently, where the probability estimation is incrementally updated after each seed selection.

On this basis, we propose our SCORE algorithm to solve the MCSS-PCG problem with theoretical guarantee. Different from our solution to MCSS-ECG, SCORE only needs one collection of RR-sets

**Procedure** VEST($u$, &$cnt$, &$covered$, &$coverage$, $\mathcal{M}$, $T$)

**Input:** seed $u^*$, count $cnt$, arrays $covered[\cdot][\cdot]$, $coverage[\cdot]$, MRR-sets $\mathcal{M}$, threshold $T$.

```
1  for i ∈ [1, κ] do
2      if coverage[i]/θ >= T then
3          continue;
4      else
5          for j ∈ [1, θ] do
6              if covered[i][j] == True then
7                  continue;
8              else
9                  if u* ∈ R_{i,j} then
10                     covered[i][j] ← True;
11                     coverage[i] ← coverage[i] + 1;
12                     if coverage[i]/θ >= T then
13                         cnt ← cnt + 1;
14                         break;
```

$\mathcal{R}$ to select the seeds. Moreover, the stopping condition becomes that the probability of $S$ reaching $\eta$ is estimated to be no less than $p + \lambda$. More implementation details could be found in Alg. 4.

Specifically, we first initialize the seed set $S$, RR-set collection $\mathcal{R}$, and MRR-set collection $\mathcal{M}$. Then, the RR-set collection $\mathcal{R}$ is filled with $\theta_e$ RR-sets, the same as the MCSS-ECG case. Following the analysis in Section 4.2, we generate $\kappa$ MRR-sets, each of which contains $\theta$ cRR-sets, resulting in the estimation error $\lambda$ and failure probability $\delta_1 + \delta_2$. Then, we introduce some global variables to track the states of MRR-sets. For each cRR-set $cRR_{i,j}$ in the MRR-set $\mathcal{M}_i$, its state of being covered by the current seed set or not is recorded by the array $covered[i][j]$, corresponding to the r.v. $Y_{i,j}$ in Eq. (8), which takes value `True` if $cRR_{i,j}$ has been covered and takes value `False` otherwise. Further, the number of cRR-sets covered by the seed set in $\mathcal{M}_i$ is recorded by the array $coverage[i]$. Moreover, if the coverage of $S$ in $\mathcal{M}_i$ reaches $\theta(\eta/n + \tau)$, then the indicator function in Eq. (8) takes value 1. Finally, we apply a counter $cnt$ to record the number of MRR-sets that satisfy the condition. Accordingly, we can estimate the probability $\mathbb{P}[I(S) \geq \eta]$ with $cnt/\kappa$ by Eq. (8).

Now, we are ready to select the seed users. Each time, we select the user $u^*$ with the largest influence-to-cost ratio as the seed, and insert it into the seed set $S$. Then, the values of $cnt$, $coverage[\cdot]$, and $covered[\cdot][\cdot]$ need to be updated due to the new seed $u^*$. To be efficient, we apply `VEST` to update the states of MRR-sets and cRR-sets which contain $u^*$. Specifically, for each MRR-set $\mathcal{M}_i$, we first examine the coverage of $S$ on it (i.e., $coverage[i]$) and take corresponding actions as follows.

If more than $\theta(\eta/n + \tau)$ cRR-sets in $\mathcal{M}_i$ have already been covered by $S$ (Line 2), we just proceed to the next MRR-set, without updating the states of its cRR-sets or the value of $cnt$. Because an MRR-set with coverage over $\theta(\eta/n + \tau)$ and the cRR-sets therein have already contributed to increasing the value of $cnt$ (Lines 12-13). Duplicated addition to $cnt$ would make the estimation not accurate any more.

If $coverage[i] < \theta(\eta/n + \tau)$, then we delve into the states of the cRR-sets. For each cRR-set $cRR_{i,j}$ in $\mathcal{M}_i$, if it has already been covered by previous seeds, we just skip this cRR-set for the next one, since the new seed $u^*$ would not change its state of being covered. If the cRR-set has not been covered, we next check whether it is covered by the new seed $u^*$. If $u^* \in cRR_{i,j}$ (Line 9), then $cRR_{i,j}$ will be covered by $u^*$. Meanwhile, the state of $cRR_{i,j}$ (i.e., $coverage[i][j]$) needs to be updated as `True`, and the coverage of $S$ in $\mathcal{M}_i$ is also increased by 1. Since the value of coverage is changed,

we have to further examine whether the new coverage exceeds $\theta(\eta/n + \tau)$. If that is the case, the value of *cnt* should also be added by 1. Finally, the new probability is estimated as $cnt/\kappa$, according to Eq. (8). If the estimated probability is still smaller than $p + \lambda$, we continue to select seeds and make new estimations until the condition is met.

Next, we build the approximation ratio of SCORE based on our results in MCSS-ECG. Specifically, we focus on deriving a practical ratio based on Lemma 3.7. To be concise, we will succeed the notations in MCSS-ECG, while they are defined under the condition that the influence will reach $\eta$ with probability at least $p$, such as $S^o, S_k$. To begin with, we measure the gap between $\eta$ and $\mathbb{I}(S^o)$ by $a = \max\{\eta - \mathbb{I}(S^o), 0\}$, and the gap between $\eta$ and $\mathbb{I}(S_{k-1})$ by $a' = \max\{\mathbb{I}(S_{k-1}) - \eta, 0\}$, which are upper bounded by Lemma 4.9.

Lemma 4.9. *The upper bounds of $a$ and $a'$ are*

$$a \leq (1-p)\eta, \qquad a' \leq pn + (1-p)\eta + 1. \tag{9}$$

Note that SCORE and CLEAR select seed users with the same idea, and the approximation ratio of CLEAR is already available in Lemma 3.7. Thus, we can bound the cost of SCORE reaching an expected influence of a smaller value $\eta - a$. Further, by the definition of $a'$, the expected influence of $S_{k-1}$ satisfies $\mathbb{I}(S_{k-1}) \leq \eta + a'$. Synthesizing the two observations, we can bound the total cost of SCORE by adding the cost to reach $\eta + a'$ from $\eta - a$ and the cost to reach $\eta - a$ from 0. With this idea, we derive Theorem 4.10, where $S_l$ is the minimum seed set selected by CLEAR whose expected influence is at least $\eta - a$, i.e., $\hat{\Gamma}(S_l) \geq (1+\epsilon)(\eta - a)$. Further, $S^*$ is the optimal seed set which achieves the expected influence of $\eta - a$ with the minimum cost, i.e., $S^* = \arg\min_{\{\Gamma(S) \geq \eta - a\}} c(S)$. Moreover, $c^{\perp}(S^*)$ is the lower bound of the optimal cost $c(S^*)$ derived by Lemma 3.7.

Theorem 4.10. *The cost of the seed set $S$ selected by* SCORE *approximates that of the optimal seed set $S^o$ in the following form*

$$c(S_k) \leq \frac{c(S_l)}{c^{\perp}(S^*)} c(S^o) + \frac{(a + a')n\rho}{n - (\eta + a')} + 2. \tag{10}$$

## 5 RELATED WORKS

The MCSS problem is dual to the classic influence maximization (IM) problem, sharing many properties with IM, e.g., the NP-hardness and the inapproximability. As a derivative of IM, the MCSS problem is formulated after IM and first studied by Long et al. They prove that the influence function is submodular and propose the first approximation algorithm for MCSS [31]. After that, a large body of works are carried out to advance the research of MCSS from various aspects [7, 10, 14, 28, 36, 37, 39, 40, 49, 51].

Initially, researchers are mainly concerned about the basic problem MCSS-ECG. Inspired by [42], Goyal et al. [18] adopt the greedy idea and propose a bi-criteria approximation algorithm. Specifically, to derive a bounded approximation ratio $\ln \eta/\epsilon$, they allow a small shortfall $\epsilon > 0$ in reaching the influence threshold. Later, Kuhnle et al. consider the case where the influence function is not strictly submodular and the influence oracle is not available [27]. As a solution, they also propose a bi-criteria approximation algorithm, whose approximation ratio however is worse than [18]. Notably, Zhu et al. eliminate the shortfall in reaching the influence threshold [52]. Their algorithm could provide an approximation ratio when an influence oracle exists, which however is not the case in practice.

The above works often suffer from inefficient and inaccurate influence estimation in their algorithm implementation. To circumvent this issue, Han et al. apply the well-known RR-sets for influence estimation [20]. On this basis, a series of algorithms are proposed with performance guarantee. Later, Hong et al. [22] develop the competitive reverse reachable sets based on RR-sets

to solve the MCSS problem under competitive influence propagation. Although [20] and [22] solve MCSS in a relatively efficient way, their algorithms still could not reach the influence threshold. Differently, Zhang et al. design an algorithm to compensate for the influence shortfall under uniform costs and very small influence estimation error [50]. Nevertheless, the case of fine-grained non-uniform costs and practical estimation error still needs further investigation.

Besides MCSS-ECG, the MCSS-PCG problem is also considered in [50]. Zhang et al. propose the MinSeed algorithm for solution, which is quite time-consuming, since the Monte Carlo method is adopted to evaluate the probability of current seeds reaching $\eta$. The theoretical performance of MinSeed is derived based on the results in MCSS-ECG. Due to the dependency on MCSS-ECG, their analysis is applicable only when the estimation error on influence is very small. Following [50], Jia et al. instantiate influence estimation with their proposed technique in the Monte Carlo fashion, which samples many possible worlds and counts the paths from seeds to a user to estimate the influence [24]. An advantage of their approach may be that the seeds' influence and their probability of reaching $\eta$ could be estimated at the same time.

Despite MCSS-ECG and MCSS-PCG , the MCSS problem is also formulated in an adaptive way. Vaswani et al. are the precursors to study the adaptive seed minimization problem [47]. They first derive a relationship between the optimal policy and the greedy policy. Applying this relationship and allowing a shortfall in reaching $\eta$, they derive an approximation ratio for their adaptive greedy policy. Later, Tang et al., argue that the influence function should be truncated, for which a new influence estimation technique named mRR-sets is proposed [43]. Drawing the theoretical results from [17], the authors show that their ASTI algorithm could achieve an approximation ratio of the form $(\ln \eta + 1)^2$.

ASTI could always solve the MCSS-PCG problem, due to its adaptive nature. However, the adaptivity also leads to its two deficiencies in solving MCSS-PCG, besides the scalability issue discussed in Section 2. First, ASTI needs to observe the diffusion from previous seeds, while it is often hard to know whether a user has adopted the product in practice. Moreover, much time is needed to finish the observation of diffusion, since the actual influence dissemination between users is not completed within milliseconds like computer simulation. With a long implementation time, ASTI could not accumulate enough engagements within short time so as to be noticed by the trending algorithms of social platforms. Second, the probability provided by ASTI is fixed, which may exceed the expectation of marketers too much and incur extra costs, thus is not flexible in meeting the requirement of MCSS-PCG.

The implementation of the above works relies on the estimation of influence and probability. Cohen proposes a sketch-based method SKIM to estimate user influence in [6]. The main idea of SKIM is to build numerous min-hash sketches based on possible worlds (i.e., realizations). Specifically, given a possible world, SKIM has to traverse the whole possible world to build sketches for each node, which is quite time-consuming. Further, in SKIM, to provide an approximation guarantee, numerous possible worlds have to be generated and each possible world has to be traversed in the above way to build sketches, resulting in high computation cost. In contrast, RR-sets based methods only require the construction of a partial possible world, a subgraph of nodes reachable from the root. Existing works, such as [2, 13, 35], have also validated that SKIM is not as efficient as RR-sets based methods. Thus, in this paper, we propose `MRR` based on RR-sets instead of sketches.

To summarize, the MCSS-PCG problem has not been well solved yet. There is still a lack of efficient methods for probability estimation, and a lack of consideration for non-uniform user costs. Moreover, the MCSS-ECG problem, which MCSS-PCG often relies on, also needs to eliminate the influence shortfall and relax the constraint on the error of influence estimation. Thus, we carry out this work to address these challenges with our `MRR`, `CLEAR`, and `SCORE` algorithms.

Table 2. Datasets. ($K = 10^3, M = 10^6, B = 10^9$)

| Name | $n$ | $m$ | Avg Degree | Type |
|---|---|---|---|---|
| *DBLP* | 655K | 2.0M | 6.1 | undirected |
| *Pokec* | 1.6M | 30.6M | 37.5 | directed |
| *Livejournal* | 4.8M | 69.0M | 28.5 | directed |
| *Friendster* | 65.6M | 1.8B | 27.5 | undirected |

## 6 EXPERIMENTS

In this section, we experimentally evaluate our algorithms against its competitors on real social networks. Specifically, we aim to answer the following questions for MCSS-PCG and MCSS-ECG.

**Q1.** Do the algorithms provide feasible solutions?
**Q2.** What is the scalability of each algorithm?
**Q3.** What is the total cost required by each algorithm?

A feasible solution for MCSS-PCG (resp. MCSS-ECG) means the output seed set $S$ can activate at least $\eta$ users with probability at least $p$ (resp. achieve an expected influence no less than $\eta$). All experiments are performed on a Linux server with an Intel Xeon 2.7GHz CPU and 400GB memory. Moreover, all codes are implemented in C++ and compiled by g++ under -O3 optimization.

### 6.1 Experimental Settings

**Datasets.** The experiments are carried out on four datasets (i.e., DBLP, Pokec, Livejournal, and Friendster), which are available in [29]. Their statistic information is summarized in Table 2. As can be seen, the dataset Friendster contains over 65 million users and 1.8 billion edges, making it the largest dataset ever tested for MCSS. Note that the influence diffusion is directed under the IC model. Thus, for undirected networks DBLP and Friendster, we transform each of their edges $(u, v)$ into two directed edges $(u, v)$ and $(v, u)$.

**Parameters.** For the IC model, like most works on IM and MCSS [3, 5, 12, 25], the diffusion probability of each edge $(u, v)$ is set to be $\frac{1}{d_{\text{in}}(v)}$. Following [19, 20, 30, 34], the cost of each user is sampled from $(0, 1]$ uniformly at random. The influence threshold $\eta$ is varied from 10% to 20% of the number of network users with a step of 2%, and the probability of reaching $\eta$ is required to be 0.6. Further, the probability estimation error $\lambda$ of our MRR-sets as well as the Monte Carlo method is set to be 0.07, and the failure probability $\delta$ is set to be 0.01. The influence estimation error $\epsilon_e$ of RR-sets is 0.1, and the failure probability is also 0.01. Finally, we simulate 10,000 diffusion processes from each seed set output by the algorithms to estimate its expected influence and the probability that the influence exceeds $\eta$.

**Algorithms.** For a comprehensive comparison, seven algorithms are evaluated in experiments, including five baselines (BCGC, TEGC, ASTI, MinSeed, GRR) and our two algorithms `CLEAR` and `SCORE`. The detailed settings of these algorithms are described as follows.

In BCGC and TEGC, a shortfall $\alpha$ is needed to decide their gap to the threshold $\eta$. To help them approach $\eta$, we set $\alpha$ to a moderately small value 0.02, instead of 0.2 in their work. Note that, the number of RR-sets demanded by BCGC and TEGC is as large as $10^{10}$ even if in the smallest dataset DBLP, and thus could not be generated within reasonable time. Therefore, we follow their empirical setting in [20], which is more efficient but provides no performance guarantee. That is, when calculating the quantity of RR-sets with the union bound, the number of unqualified seed sets is set to be a fixed value $n^8$.

The adaptive algorithm ASTI is found to be highly time-consuming when we adopt the same influence estimation error 0.1 as other algorithms. Thus, to obtain the results of ASTI, we relax its estimation error to be 0.5. To further boost it, we allow it to select 4 seeds in each round, slightly

sacrificing the approximation ratio. To eliminate the randomness of adaptive algorithms, ASTI is repeated 10 times to obtain an average.

MinSeed [50] is dedicatedly designed for MCSS-PCG. As MinSeed also takes much time to complete, due to its frequent invocation of the Monte Carlo method for probability estimation, we accelerate it by making probability estimation after selecting every 10 seeds, incurring a little more seed cost though.

The GRR technique is also implemented for comparison with our MRR-sets. Specifically, GRR adopts the same framework as SCORE to select the seeds. To estimate the probability $\mathbb{P}[I(S) \geq \eta]$, GRR generates $\kappa$ groups of RR-sets, each group containing $\theta$ *independent* RR-sets, following a similar structure of MRR. On this basis, GRR estimates the probability in the way introduced at the beginning of Section 4.

In our algorithm CLEAR, in addition to the aforementioned parameters, we further specify the validation error $\epsilon_v$ to be 0.01, and the minimum influence $\xi$ to be $0.002\eta$. When estimating the probability with the MRR technique in SCORE, we set the failure probabilities to be $\delta_1 = 0.01/3$ and $\delta_2 = 0.02/3$, resulting in the same total failure probability of 0.01 as the Monte Carlo method. Finally, the estimation error of each MRR-set is required to be $\tau = 0.01$.

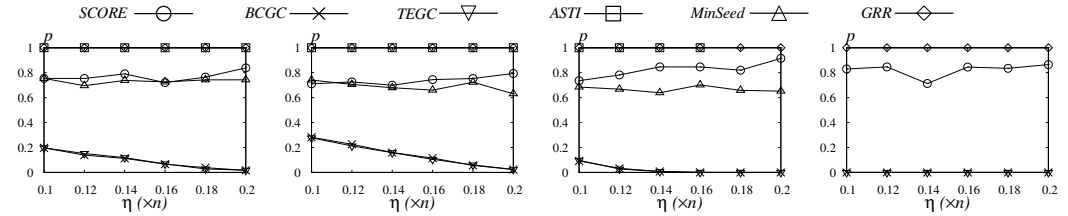
### 6.2 Results under MCSS-PCG

For the main problem MCSS-PCG, our algorithm SCORE is evaluated against all the five baselines. The results to the three research questions are presented as follows.

**Probability Attainment.** To answer the research question Q1 in MCSS-PCG, we first examine the probability that the seed set returned by each algorithm reaches the influence threshold $\eta$. The experimental results are presented in Fig. 2. Note that an algorithm is considered to be feasible for the MCSS-PCG problem, only when its output seed set reaches influence $\eta$ with a probability no less than $p = 0.6$.

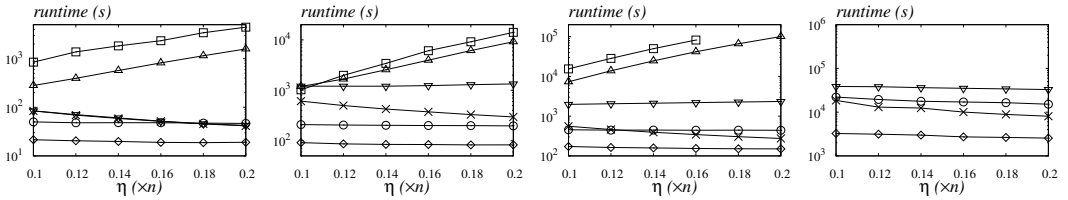
As can be seen, our algorithm SCORE and three of the baselines (i.e., MinSeed, ASTI, and GRR) always provide a feasible solution to MCSS-PCG, i.e., achieving a probability no less than 0.6. Our algorithm SCORE and MinSeed are able to attain the probability since they apply their probability estimation methods to make sure that the probability is achieved when the seed selection stops. In the adaptive case, ASTI will continue selecting seeds until the threshold is observed to be reached. Thus, ASTI will always reach $\eta$ with probability 1, producing feasible solutions for any MCSS-PCG problem. Note that there are no data points for ASTI in LiveJournal (at $\eta > 0.16$) or Friendster, and no data points for MinSeed in Friendster, as they failed to terminate within the time limit of $10^5$ seconds. GRR consistently achieves $\eta$ with a probability of 1. This occurs because GRR tends to underestimate the probability of reaching $\eta$, and as a result, it selects more seeds than necessary. For detailed explanations, please refer to the example in Appendix D of [1]. Moreover, BCGC and TEGC return seed sets that reach influence $\eta$ with a probability much smaller than 0.6 in all datasets. For some values of $\eta$, the probabilities even approach 0, especially in large networks Livejournal and Friendster. The reason is that the parameters in BCGC and TEGC are set heuristically, as settings derived from their theoretical analysis would lead to prohibitive computational costs. As a result, there is no guarantee on their output solutions.

**Running Time.** To answer Q2 in MCSS-PCG, we present the running time of the algorithms, as shown in Fig. 3. Compared with MinSeed and ASTI which can provide feasible solutions, our algorithm SCORE is at least one order of magnitude faster. Especially, MinSeed (resp. ASTI) takes 200x (resp. 180x) more time than SCORE in Livejournal at $\eta = 0.2$ (resp. $\eta = 0.16$). The inefficiency of MinSeed results from its requirement to invoke hundreds of diffusion simulations for each probability estimation during seed selection. On the other hand, the reason of ASTI having the



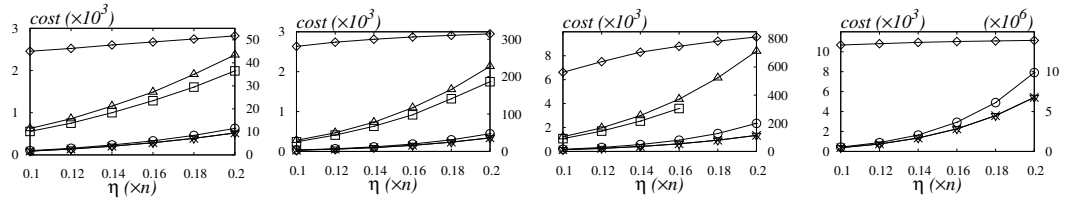

(a) DBLP (b) Pokec (c) Livejournal (d) Friendster

Fig. 2. Varying $\eta$: probability attainment of each algorithm in MCSS-PCG.



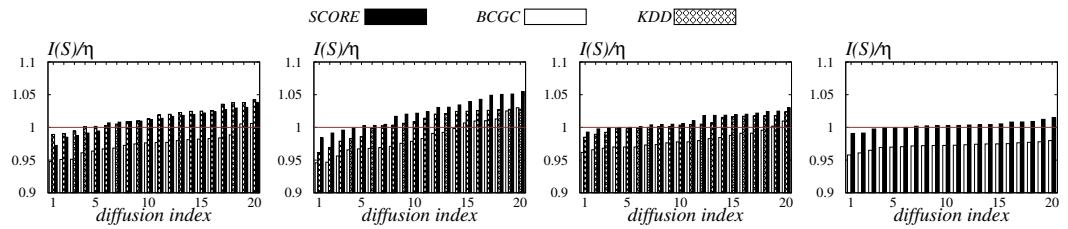

(a) DBLP (b) Pokec (c) Livejournal (d) Friendster

Fig. 3. Varying $\eta$: running time of each algorithm in MCSS-PCG.




(a) DBLP (b) Pokec (c) Livejournal (d) Friendster

Fig. 4. Varying $\eta$: seed cost of each algorithm in MCSS-PCG.




(a) DBLP (b) Pokec (c) Livejournal (d) Friendster

Fig. 5. Varying diffusion: spread results of 20 times of random diffusion.

largest running time is two-fold. First, due to its adaptive nature, brand new mRR-sets have to be generated after each round of seed selection and influence diffusion. Second, the influence of the 4 seeds in each round is relatively small, which needs a large number of mRR-sets for estimation. GRR takes less running time than SCORE. To explain, the RR-sets in GRR are generated independently, thereby avoiding the overhead involved in examining and maintaining edge states, which however is necessary for MRR. Note that BCGC achieves a running time comparable to our SCORE. However, as shown in Fig. 2, it always fails to produce feasible solutions for MCSS-PCG, rendering it substantially unattractive.

**Seed Cost.** To answer Q3 in MCSS-PCG, we report the costs of the seed sets returned by the algorithms, as shown in Fig. 4. Compared with other algorithms (i.e., MinSeed, ASTI, and GRR)

that produce feasible solutions to MCSS-PCG, our algorithm SCORE consumes the smallest cost. ASTI spends much larger cost than SCORE, since it always has to reach $\eta$ with certainty, resulting in larger expected influence and thus larger cost. Meanwhile, when selecting the seed users, ASTI focuses on selecting users with the largest influence, without considering the cost that would be incurred. The cost of MinSeed is even larger than ASTI, since MinSeed is not an adaptive algorithm and thus could not make sensible seed selection based on previous diffusion results, while the costs of users are also neglected like ASTI. Moreover, the costs of BCGC and TEGC are smaller than other algorithms, since they could not reach the influence threshold with the desired probability, and the gap is significant. Figs. 2–3 show that GRR constantly produces feasible solutions, and runs faster than our SCORE. Note that the right vertical axis in each subfigure of Fig. 4 is especially created to incorporate the cost of GRR. As can be seen, GRR incurs extremely higher seed costs to solve MCSS-ECG, which is at least 21.8 times greater than that of MinSeed. This discrepancy can be attributed to GRR's inaccurate and pessimistic probability estimation, leading to the inclusion of far more seeds than necessary. Therefore, despite GRR's efficiency in providing feasible solutions, its prohibitive cost makes it impractical for real-world applications.

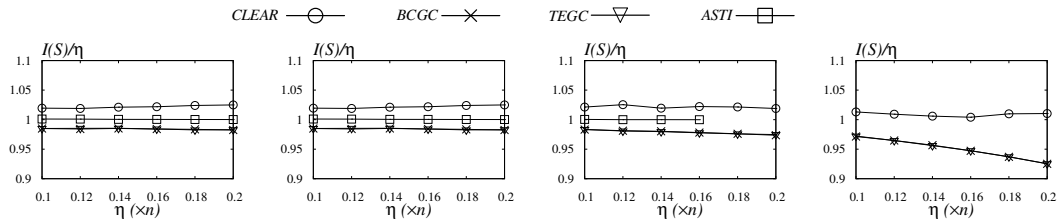
**Spread Visualization.** To provide an intuitive answer to Q1, in Fig. 5 we visualize the influence spread of three algorithms (SCORE, MinSeed, and BCGC) at $\eta = 0.1n$. ASTI is not selected, since it always reach $\eta$ with certainty. TEGC is not visualized either, due to its similar influence to BCGC. Specifically, we carry out 20 times of random diffusion from the seed sets of the three algorithms. According to our setting of $p = 0.6$, an algorithm which reaches $\eta$ in more than 12 samples is considered to be feasible. As can be seen, SCORE and MinSeed have over 12 samples reaching $\eta$ in all datasets, satisfying the requirement of MCSS-PCG. For our algorithm SCORE, the minimum number of diffusion samples that exceeds $\eta$ is 14, obtained in Friendster; and the result for MinSeed is also 14, derived in Pokec and Livejournal. Note that, there is also no result for MinSeed in Friendster, due to its excessive running time. On the other hand, the number of diffusion samples of BCGC that exceeds $\eta$ is significantly smaller than BCGC and MinSeed, where the maximum number of diffusion samples that reach $\eta$ is 6, derived in Pokec; and the minimum number is as small as 0, derived in Friendster. Thus, BCGC is significantly inferior to SCORE, and cannot provide feasible solutions to MCSS-PCG.

### 6.3 Results under MCSS-ECG

In MCSS-ECG, we compare our algorithm CLEAR with the three baselines (BCGC, TEGC, and ASTI) designed for MCSS-ECG.

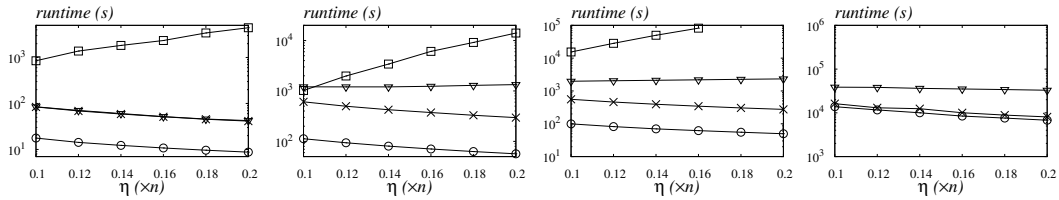
**Threshold Attainment.** In MCSS-ECG, an algorithm is feasible if it reaches the influence threshold $\eta$ in expectation. Thus, to answer Q1 in MCSS-ECG, we present the expected influence achieved by the algorithms in Fig. 6. As can be seen, only our algorithm CLEAR and the adaptive algorithm ASTI always reach $\eta$ in the four datasets. The attainment of CLEAR is due to the introduction of an independent collection of validation RR-sets, which ensures that the seeds could reach $\eta$ w.h.p. ASTI could reach the threshold without much influence surplus, since it stops seed selection when the observed influence spread just reaches $\eta$. The other two algorithms BCGC and TEGC constantly fail to reach $\eta$, even if we have reduced their gaps by adopting a small $\alpha$. Note that, the shortfall $\alpha$ cannot be set to be 0, since the approximation ratio $(1 + \ln \frac{3-\alpha}{\alpha})$ would be invalid when $\alpha = 0$ [20]. Such a non-zero shortfall is also the cause of their failure. Moreover, their gap to $\eta$ is even broader with the increase of $\eta$ in Friendster, up to 7.4%, since Friendster demands a large number of RR-sets for accurate seed selection, which cannot be satisfied by the empirical setting in [20].

**Running Time.** To answer Q2, we measure the efficiency of the algorithms by their running time. From Fig. 7, we observe that the running time of our algorithm CLEAR is the smallest. While BCGC



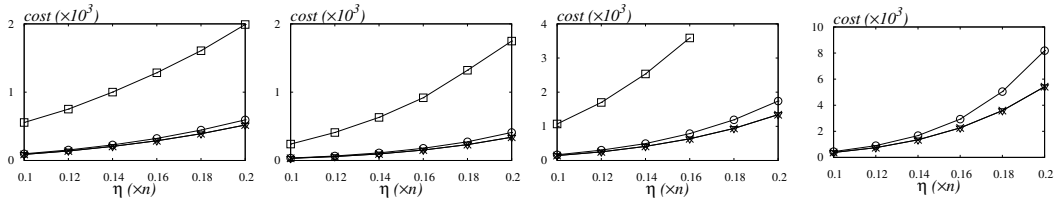

(a) DBLP (b) Pokec (c) Livejournal (d) Friendster

Fig. 6. Varying $\eta$: threshold attainment of each algorithm in MCSS-ECG.




(a) DBLP (b) Pokec (c) Livejournal (d) Friendster

Fig. 7. Varying $\eta$: running time of each algorithm in MCSS-ECG.



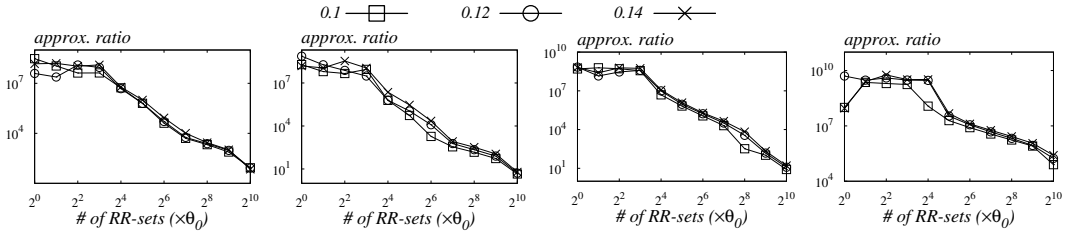

(a) DBLP (b) Pokec (c) Livejournal (d) Friendster

Fig. 8. Varying $\eta$: seed costs of each algorithm in MCSS-ECG.

(resp. TEGC) requires as much as 5.5 (resp. 47.1) times longer time than CLEAR in Livejournal at $\eta = 0.2$. The large running time of BCGC and TEGC is mainly due to their over pessimistic analysis of the RR-sets quantity, which is required to ensure an estimation error for any possible seed set. Moreover, the running time of ASTI is the largest, over 700 times larger than CLEAR, due to the same reason in MCSS-PCG. Note that, even if another collection of RR-sets $\mathcal{R}_2$ is generated in CLEAR, the running time does not increase much, since $\mathcal{R}_2$ only needs to ensure the accuracy of the final influence, which is often a large value. As a result, not many RR-sets need to be generated according to Lemma 3.1.

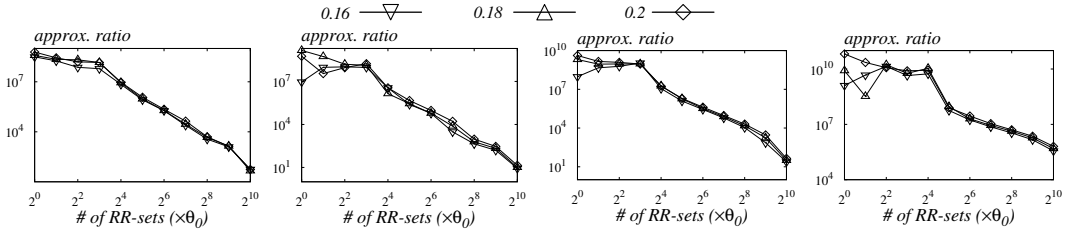
**Seed Cost.** To answer Q3 in MCSS-ECG, we present the cost of each algorithm in Fig. 8. It is shown that the cost of our algorithm CLEAR is significantly smaller than the adaptive algorithm ASTI, where the reason is similar to the case of MCSS-PCG. The costs of BCGC and TEGC are slightly smaller than CLEAR, since they do not need to reach the influence threshold like CLEAR. Moreover, their excessive number of RR-sets may also provide more accurate influence estimation.

**Approximation Ratio.** Lemma 3.7 allows us to derive an explicit approximation ratio of our algorithm CLEAR. Further, combined with Lemma 3.6, Lemma 3.7 implies that CLEAR can provide a performance guarantee under any number of RR-sets. Thus, to verify these Lemmas, we vary the number of RR-sets and report the approximation ratio of CLEAR in Fig. 9 and Fig. 10. Given the original number of RR-sets $\theta_e$ adopted by Alg. 1, we set the initial number of RR-sets to be $\theta_0 = \theta_e \times 2^{-10}$. Then, we increase the RR-sets from $\theta_0$ to $\theta_0 \times 2^{10}$ with a common ratio of 2. As




(a) DBLP (b) Pokec (c) Livejournal (d) Friendster

Fig. 9. Varying RR-sets: approximation ratio of CLEAR at $\eta/n = 0.1, 0.12$ and 0.14.




(a) DBLP (b) Pokec (c) Livejournal (d) Friendster

Fig. 10. Varying RR-sets: approximation ratio of CLEAR at $\eta/n = 0.16, 0.18$ and 0.2.

can be seen, the approximation ratio of CLEAR is generally decreasing with the increase of RR-sets in all datasets. Thus, the performance of CLEAR is becoming better when the number of RR-sets increases, since we are minimizing a function. The reason may lie in that with more RR-sets, the influence estimation is more accurate, and the selection of seeds would prefer users whose actual influence is large, leading to smaller cost to reach $\eta$ and thus its ratio to the optimal cost is smaller.

### 6.4 Experiments under Alternative Models

To evaluate the performance of our algorithms comprehensively, we further conduct experiments under the LT model and degree-based cost model. Specifically, in the LT model, the weight of each edge $(u, v)$ is also set to be $\frac{1}{d_{\text{in}}(u)}$, just like [3, 43–45], and the threshold of each user is selected from the interval $[0, 1]$ uniformly at random. In terms of the degree-based cost model, we follow [3, 9, 19] to set the cost of each user $u$ to be $c(u) = 0.01\,(d_{\text{in}}(u) + 1)$, where the coefficient 0.01 is in line with our investigation in Section 1 and the degree is added by 1 to avoid zero cost. Due to space limitations, the results and corresponding explanations are deferred to Appendix E in [1].

## 7 CONCLUSION

In this paper, we solve the MCSS-PCG problem under non-uniform costs, by designing efficient algorithms with theoretical guarantees. First, we tackle the MCSS-ECG problem with our CLEAR algorithm, allowing no influence shortfall. Then, the STAR method is proposed to derive an explicit approximation ratio of CLEAR. Next, the MRR technique is devised for efficient probability estimation. On this basis, we solve the MCSS-PCG problem with our SCORE. Finally, extensive experiments on real social networks validate the effectiveness and efficiency of our algorithms.

## ACKNOWLEDGMENTS

Sibo Wang is supported by the RGC GRF grant (No. 14217322), Hong Kong ITC ITF grant (No. MRP/071/20X), and Tencent Rhino-Bird Focused Research Grant. Chen Feng is supported by Hong Kong ITC RTH grant (No. PiH/012/22)

## REFERENCES


[1] 2024. Technical Report. https://github.com/Planet-B612/MCSS.

[2] Akhil Arora, Sainyam Galhotra, and Sayan Ranu. 2017. Debunking the myths of influence maximization: An in-depth benchmarking study. In *SIGMOD*. 651–666.

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

[4] Christian Borgs, Michael Brautbar, Jennifer Chayes, and Brendan Lucier. 2014. Maximizing social influence in nearly optimal time. In *SODA*. 946–957.

[5] Wei Chen, Yifei Yuan, and Li Zhang. 2010. Scalable influence maximization in social networks under the linear threshold model. In *ICDM*. IEEE, 88–97.

[6] Edith Cohen, Daniel Delling, Thomas Pajor, and Renato F Werneck. 2014. Sketch-based influence maximization and computation: Scaling up with guarantees. In *CIKM*. 629–638.

[7] Victoria Crawford, Alan Kuhnle, and My Thai. 2019. Submodular cost submodular cover with an approximate oracle. In *ICML*. 1426–1435.

[8] Artem Dogtiev. 2023. Influencer Marketing Costs. https://www.businessofapps.com/marketplace/influencer-marketing/research/influencer-marketing-costs/.

[9] Nan Du, Yingyu Liang, Maria-Florina Balcan, Manuel Gomez-Rodriguez, Hongyuan Zha, and Le Song. 2017. Scalable Influence Maximization for Multiple Products in Continuous-Time Diffusion Networks. *J. Mach. Learn. Res.* 18, 2 (2017), 1–45.

[10] Lidan Fan, Zaixin Lu, Weili Wu, Bhavani Thuraisingham, Huan Ma, and Yuanjun Bi. 2013. Least cost rumor blocking in social networks. In *ICDCS*. 540–549.

[11] Uriel Feige. 1998. A threshold of ln n for approximating set cover. *J. ACM* 45, 4 (1998), 634–652.

[12] Chen Feng, Luoyi Fu, Bo Jiang, Haisong Zhang, Xinbing Wang, Feilong Tang, and Guihai Chen. 2020. Neighborhood matters: Influence maximization in social networks with limited access. *IEEE Trans. Knowl. Data Eng.* 34, 6 (2020), 2844–2859.

[13] Sainyam Galhotra, Akhil Arora, and Shourya Roy. 2016. Holistic influence maximization: Combining scalability and efficiency with opinion-aware models. In *SIGMOD*. 743–758.

[14] Guoju Gao, Mingjun Xiao, Jie Wu, He Huang, and Guoliang Chen. 2018. Minimum cost seed selection for multiple influences diffusion in communities. In *MASS*. 263–271.

[15] Malcolm Gladwell. 2006. *The Tipping Point: How Little Things Can Make a Big Difference*. Hachette UK.

[16] GOAT. 2020. Case Study: DELL. https://goatagency.com/case/dell-3/.

[17] Daniel Golovin and Andreas Krause. 2011. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *J. Artif. Intell. Res.* 42 (2011), 427–486.

[18] Amit Goyal, Francesco Bonchi, Laks VS Lakshmanan, and Suresh Venkatasubramanian. 2013. On minimizing budget and time in influence propagation over social networks. *Social Netw. Anal. Mining* 3 (2013), 179–192.

[19] Qintian Guo, Chen Feng, Fangyuan Zhang, and Sibo Wang. 2023. Efficient Algorithm for Budgeted Adaptive Influence Maximization: An Incremental RR-set Update Approach. *SIGMOD* 1, 3 (2023), 1–26.

[20] Kai Han, Yuntian He, Keke Huang, Xiaokui Xiao, Shaojie Tang, Jingxin Xu, and Liusheng Huang. 2019. Best bang for the buck: Cost-effective seed selection for online social networks. *IEEE Trans. Knowl. Data Eng.* 32, 12 (2019), 2297–2309.

[21] Kai Han, Keke Huang, Xiaokui Xiao, Jing Tang, Aixin Sun, and Xueyan Tang. 2018. Efficient algorithms for adaptive influence maximization. *VLDB* 11, 9 (2018), 1029–1040.

[22] Wenjing Hong, Chao Qian, and Ke Tang. 2020. Efficient minimum cost seed selection with theoretical guarantees for competitive influence maximization. *IEEE Transactions on Cybernetics* 51, 12 (2020), 6091–6104.

[23] Shixun Huang, Junhao Gan, Zhifeng Bao, and Wenqing Lin. 2023. Managing Conflicting Interests of Stakeholders in Influencer Marketing. *SIGMOD* 1, 1 (2023), 1–27.

[24] Su Jia, Ling Chen, Yixin Chen, Bin Li, and Wei Liu. 2021. Minimizing the seed set cost for influence spreading with the probabilistic guarantee. *Knowledge-Based Systems* 216 (2021), 106797.

[25] David Kempe, Jon Kleinberg, and Éva Tardos. 2003. Maximizing the spread of influence through a social network. In *KDD*. 137–146.

[26] Larry Kim. 2019. 9 Ways To Improve Your Chances Of Getting Viral Marketing Campaigns. https://serpstat.com/blog/9-ways-to-improve-your-chances-of-getting-viral-marketing-campaigns/.

[27] Alan Kuhnle, Tianyi Pan, Md Abdul Alim, and My T Thai. 2017. Scalable bicriteria algorithms for the threshold activation problem in online social networks. In *INFOCOM*. 1–9.

[28] Zhuo Qi Lee and Wen-Jing Hsu. 2015. Minimizing Seed Selection for Disseminating News with Probabilistic Coverage Guarantee. In *International Conference on Electronic Commerce*. 1–8.

[29] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.
[30] Xiang Li, J David Smith, Thang N Dinh, and My T Thai. 2019. Tiptop:(almost) exact solutions for influence maximization in billion-scale networks. *IEEE/ACM Transactions on Networking* 27, 2 (2019), 649–661.
[31] Cheng Long and Raymond Chi-Wing Wong. 2011. Minimizing seed set for viral marketing. In *ICDM*. 427–436.
[32] Christoph E Mandl. 2019. *Managing Complexity in Social Systems*. Springer.
[33] Pascal Massart. 1990. The tight constant in the Dvoretzky-Kiefer-Wolfowitz inequality. *The Annals of Probability* (1990), 1269–1283.
[34] Huy Nguyen and Rong Zheng. 2013. On budgeted influence maximization in social networks. *IEEE Journal on Selected Areas in Communications* 31, 6 (2013), 1084–1094.
[35] Hung T Nguyen, My T Thai, and Thang N Dinh. 2016. Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks. In *SIGMOD*. 695–710.
[36] Phuong NH Pham, Canh V Pham, Hieu V Duong, Trung Thanh Nguyen, and My T Thai. 2021. Groups Influence with Minimum Cost in Social Networks. In *International Conference on Computational Data and Social Networks*. 231–242.
[37] Phuong NH Pham, Canh V Pham, Hieu V Duong, Václav Snášel, and Nguyen Trung Thanh. 2023. Minimizing cost for influencing target groups in social network: A model and algorithmic approach. *Computer Communications* 212 (2023), 182–197.
[38] Whitney Rudeseal. 2023. 3 social media experts on how to go viral. https://www.stagwellmarketingcloud.com/blog/3-social-media-managers-on-how-to-go-viral.
[39] Yanchen Shi. 2019. *Algorithms for influence maximization and seed minimization*. Ph. D. Dissertation. Nanyang Technological University.
[40] Yishuo Shi, Yingli Ran, Zhao Zhang, James Willson, Guangmo Tong, and Ding-Zhu Du. 2019. Approximation algorithm for the partial set multi-cover problem. *Journal of Global Optimization* 75, 4 (2019), 1133–1146.
[41] Abhishek Singh, Niraj Dharamshi, Preethi Thimma Govarthanarajan, Premal Dattatray Samale, and Magdalini Eirinaki. 2020. The tipping point in social networks: investigating the mechanism behind viral information spreading. In *International Conference on Big Data Computing Service and Applications*. 54–61.
[42] Petr Slavík. 1997. Improved performance of the greedy algorithm for partial cover. *Inform. Process. Lett.* 64, 5 (1997), 251–254.
[43] Jing Tang, Keke Huang, Xiaokui Xiao, Laks VS Lakshmanan, Xueyan Tang, Aixin Sun, and Andrew Lim. 2019. Efficient approximation algorithms for adaptive seed minimization. In *SIGMOD*. 1096–1113.
[44] Jing Tang, Xueyan Tang, Xiaokui Xiao, and Junsong Yuan. 2018. Online Processing Algorithms for Influence Maximization. In *SIGMOD*. 991–1005.
[45] Youze Tang, Yanchen Shi, and Xiaokui Xiao. 2015. Influence maximization in near-linear time: A martingale approach. In *SIGMOD*. 1539–1554.
[46] Youze Tang, Xiaokui Xiao, and Yanchen Shi. 2014. Influence maximization: near-optimal time complexity meets practical efficiency. In *SIGMOD*. 75–86.
[47] Sharan Vaswani and Laks VS Lakshmanan. 2016. Adaptive influence maximization in social networks: Why commit when you can adapt? *arXiv preprint arXiv:1604.08171* (2016).
[48] Yexin Wang, Zhi Yang, Junqi Liu, Wentao Zhang, and Bin Cui. 2023. Scapin: Scalable Graph Structure Perturbation by Augmented Influence Maximization. *SIGMOD* 1, 2 (2023), 1–21.
[49] Ruidong Yan, Yuqing Zhu, Deying Li, and Zilong Ye. 2019. Minimum cost seed set for threshold influence problem under competitive models. *World Wide Web* 22, 6 (2019), 2977–2996.
[50] Peng Zhang, Wei Chen, Xiaoming Sun, Yajun Wang, and Jialin Zhang. 2014. Minimizing seed set selection with probabilistic coverage guarantee in a social network. In *KDD*. 1306–1315.
[51] Jianwen Zhao and Yufei Tao. 2021. Minimum vertex augmentation. *VLDB* 14, 9 (2021), 1454–1466.
[52] Yuqing Zhu, Deying Li, and Zhao Zhang. 2016. Minimum cost seed set for competitive social influence. In *INFOCOM*. 1–9.

Received January 2024; revised April 2024; accepted May 2024