An Efficient Randomized Algorithm for Rumor Blocking in Online Social Networks

Guangmo (Amo) Tong*, Weili Wu†‡¶, Ling Guo†, Deying Li‡, Cong Liu*, Bin Liu§ and Ding-Zhu Du*

*Dept. of Computer Science, University of Texas at Dallas, USA
†Taiyuan University of Technology, China
‡Renmin University, China
§Ocean University of China, China
¶Corresponding author
{guangmo.tong, cx1131130, weiliwu, cx1137330, dzdu}@utdallas.edu
{guoling, deyingli}@ruc.edu.cn, binliu@ouc.edu.cn

Abstract—Social networks allow rapid spread of ideas and innovations while the negative information can also propagate widely. When the cascades with different opinions reaching the same user, the cascade arriving first is the most likely to be taken by the user. Therefore, once misinformation or rumor is detected, a natural containment method is to introduce a positive cascade competing against the rumor. Given a budget $k$, the rumor blocking problem asks for $k$ seed users to trigger the spread of the positive cascade such that the number of the users who are not influenced by rumor can be maximized. The prior works have shown that the rumor blocking problem can be approximated within a factor of $(1 - 1/e - \delta)$ by a classic greedy algorithm combined with Monte Carlo simulation with the running time of $O((\ln n + \ln m)/\delta^2)$, where $n$ and $m$ are the number of users and edges, respectively. Unfortunately, the Monte-Carlo-simulation-based methods are extremely time consuming and the existing algorithms either trade performance guarantees for practical efficiency or vice versa. In this paper, we present a randomized algorithm which runs in $O((\ln n \ln m)/\delta^2)$ expected time and provides a $(1 - 1/e - \delta)$-approximation with a high probability. The experimentally results on both the real-world and synthetic social networks have shown that the proposed randomized rumor blocking algorithm is much more efficient than the state-of-the-art method and it is able to find the seed nodes which are effective in limiting the spread of rumor.

I. INTRODUCTION

The tremendous advance of the Internet of things (IoT) is making the online social networks be the most common platform for communication. There have been totally 44.5 million users on Twitter and 1.4 million monthly active users on Facebook. Admittedly the online social networks are greatly beneficial, they also lead the widespread of negative information. Such negative influence, namely misinformation and rumor, has been a cause of concern as it renders the network unreliable and may cause further panic in population. For example, the misinformation of swine flu in Twitter threw the people in Texas and Kansas into panic in 2009 [1], and the endless report of Ebola in 2014 has caused unnecessary worldwide terror. Therefore, effective strategies for rumor containment are crucial for social networks and it has been a hot topic in the last decades.

In a social network, information and innovations diffuse from user to user via influence cascades where each cascade starts to spread with a set of seed users. When two cascades holding opposing views reach a certain user, the user is likely to trust the cascade arriving first. For the example of swine flu, if the international institutions like WHO would have posts clarification for swine flu, the users who have read such posts will not be influenced by the misinformation. Therefore, the most common method for rumor blocking is to generate a corresponding positive cascade that competes against the rumor. Due to the expense of deploying seed nodes, there is a budget $k$ for the positive cascade, and naturally one should select the $k$ positive seed nodes such that the number of rumor-activated users is minimized, which is referred as the least cost rumor blocking problem.

The recent study of influence diffusion process in social networks can be traced back to D. Kempe [2] where the well-known influence maximization problem is formulated. In that seminal work, two fundamental probabilistic operational models, independent cascade (IC) model and linear threshold (LT) model are developed. Based on such models, many influence related problems are then proposed and studied. The problem of rumor blocking is also considered in such models or in their variants. Most existing approaches utilize the submodularity of the objective function. That is, the number of non-rumor-activated users is a monotone increasing submodular function and therefore the classic hill-climbing algorithm provides a $(1 – 1/e)$-approximation [3]. For example, X. He et al. [4] formulate the influence blocking maximization problem and show a $(1 – 1/e)$-approximation algorithm for the competitive linear threshold model, Budak et al. [5] propose several competitive models and show a greedy algorithm with the same approximation ratio under the campaign-oblivious independent cascade model, and, Fan et al. [6] provide a $(1 – 1/e)$-approximation algorithm for the rumor blocking problem under the opportunistic one-active-one model.

Assuming that the objective function can be efficiently calculated for any input, the greedy algorithm is simple and effective for most of the submodular maximization problems. Unfortunately, for the influence related optimization problems,
the objective functions are often very complicated to compute due to the randomness of the probabilistic diffusion model. Such a scenario is first observed by W. Chen [7] where it is shown that given a seed set computing the expected influence is #P-hard. In order to circumvent such difficulty, the prior works employ the Monte-Carlo simulation to calculate the expected number of non-rumor-activated nodes. However, such a method is computationally expensive. It turns out that the greedy algorithm with Monte-Carlo simulation has the $\Omega(k \cdot m \cdot n \cdot \text{poly}(\delta^{-1}))$ time complexity to achieve a $(1 - 1/e - \delta)$ approximation ratio, and it takes several days even for small networks. With the recently analysis of influence diffusion [8]–[10], the difficulty in solving such problems has shifted from the nodes selection strategy to the calculation of the objective function. Fundamentally, it asks for a better sampling method to estimate the expected influence given the seed sets. To the best of our knowledge, there is no rumor blocking algorithm that can meet practical efficiency without sacrificing performance guarantee.

In this paper, we will show an efficient randomized algorithm for the rumor blocking problem, which is termed as the reverse-tuple based randomized (RBR) algorithm. The RBR algorithm runs in $O(k \cdot m \cdot \ln n)$ in expect and returns a $(1 - 1/e - \delta)$ approximation with a high probability. The proposed algorithm utilizes the reverse-tuple based sampling method which is more effective than the Monte-Carlo simulation used in the prior works. The reverse sampling technique is first designed by C. Borgs [8] for the influence maximization problem. In this paper, we develop a new type of sampling based on the concept of reverse-tuple (R-tuple) and show how such sampling method can be applied to the rumor blocking problem. Although both the sampling methods give the unbiased estimate, one set of Monte-Carlo simulations can only provide an estimation for a specified seed set while the samples produced by the reverse-tuple based sampling can be applied to any seed sets. The RBR algorithm can be implemented with tunable parameters and it is flexible for balancing the running time and the error probability. We experimentally evaluate the proposed algorithm on both the real-world social network and synthetic power-law networks. The experimental results show that the RBR algorithm not only produces high quality positive seed set but also takes much less time than the greedy algorithm with the Monte-Carlo simulation does. In particular, when $\delta = 0.1$ and the error probability is set as less than $1/n$ where $n$ is the number of users, the running time of the RBR algorithm is $10$ times less than that of the state-of-the-art approach. The contribution of this paper is summarized as follows:

- We design the reverse-tuple based sampling method which can be used to obtain a unbiased estimate for the objective function of the rumor blocking problem.
- Based on the new sampling technique, we present the RBR approximation algorithm which is effective and efficient for blocking rumors in the IC model.
- We evaluate the proposed algorithm via experiments done on real-world social networks and synthetic power-law networks, and show that the RBR algorithm outperforms the existing methods by a significant magnitude in terms of the running time.

The rest of the paper is organized as follows. Sec. III is devoted to the related work. The preliminaries are provided in Sec. IV. The new sampling method together with the RBR algorithm is shown in Sec. V. The experiments are presented in Sec. VI. In Sec. VII we discuss the future work and conclude.

II. RELATED WORK

In this section we will briefly survey the prior works regarding rumor controlling.

C. Budak et al. [5] are among the first who study the misinformation containment problem. In particular, they consider the multi-campaign independent cascade model and investigate the problem of identifying a subset of individuals that need to be convinced to adopt the “good” campaign so as to minimize the number of people that adopt the rumor. X. He et al. [4] and L. Fan et al. [6] further study this problem under the competitive linear threshold model and the OPOAO model, respectively. S. Li et al. [17] later formulate the $\gamma - k$ rumor restriction problem and shown a $(1 - 1/e)$-approximation. As mentioned earlier, the existing approaches are very time consuming and thus cannot handle large social networks. Recently, several heuristic methods have been proposed by different works, such as [18], [19], while they cannot provide any performance guarantee. In this paper, we aim to design the rumor blocking algorithm which is provably effective and also efficient.

Rumor source detection is another important problem for rumor controlling. The prior works primarily focus on the classic susceptible-infected-recovered (SIR) model where the nodes can be infected by rumor and may recover later. Shah et al. [15] provide a systematic study and design a rumor source estimator based upon the concept of rumor centrality. Z. Wang et al. [16] propose a unified inference framework based on the union rumor centrality, and provide explicit detection performance for degree-regular tree networks.

III. SYSTEM MODEL

In this section, we provide the preliminaries of this paper.

A. Influence Models

A social network is represented by a directed graph $G = (V, E)$ where the users are denoted by nodes and edges in $E$ show the social relationships. For a network $G$, let $E(G)$ and $V(G)$ be the edge-set and node-set of $G$, respectively. In order to spread an idea or to advertise a new product in a social network, some seed nodes are chosen to be activated to trigger the spread of influence. The diffusion process terminates when there is no user can be further activated. The two basic diffusion models are shown as follows.

Linear Threshold (LT) model. Associated with each edge $(u, v)$ there is an weight $w_{(u,v)} \in [0, 1]$ and each node $u$ has a threshold $\theta_u$, where $\sum_v w_{(v,u)} \leq 1$. For a node $u$
other than the seed nodes, \( u \) becomes active at time step \( t \) if \( \sum_{v \in A_{t-1}} u(v,u) \geq \theta_u \), where \( A_{t-1} \) is the set of the users activated at time \( t - 1 \).

**Independent Cascade (IC) Model.** Associated with each edge \((u,v)\) there is a probability \( p^G_{(u,v)} \). When node \( u \) becomes active at time \( t - 1 \), it activates each inactive neighbor \( v \) at time step \( t \) with a probability of \( p^G_{(u,v)} \). For each pair of nodes \( u \) and \( v \), \( u \) has only one chance to activate \( v \).

In this paper, we adopt the IC model to show the randomized algorithm.

### B. Rumor and Competing Cascade

Suppose there are multiple cascades each of which is generated by its own seed set. In the network, each node is initially inactive and never changes its state once activated by one cascade. Therefore, the cascade arriving first will dominate the node. In order to limit the spread of rumor, we introduce a competing cascade denoted as the positive cascade. At each time step, if a node is successfully activated by two or more neighbors belonging to different cascade, it will select the one with the highest priority. We assume that the rumor has the higher priority, because the rumor always polishes itself to be convincing. We denote by \( S_r \) and \( S_p \) the seed sets of rumor and the competing positive cascade. In this paper, we assume \( S_r \) is fixed. The diffusion process unfolds in discrete, as follows.

- Initially all the nodes are inactive.
- At time 0, nodes in \( S_r \) and \( S_p \) are activated by the rumor and the positive cascade, respectively.
- At time \( t > 0 \), each node \( u \) which is activated at \( t - 1 \) will activate each of its inactive neighbors \( v \) with a success probability of \( p^G_{(u,v)} \). If node \( v \) is successfully activated by the two cascades simultaneously at time \( t \), then \( v \) will be activated by rumor.
- The diffusion process terminates when there is no node can be further activated.

### C. The problem

Given an IC network \( G \), the seed set \( S_r \) of rumor and a integer \( k \), let \( f(S_p) \) be the expected number of nodes that are not activated by rumor when \( S_p \) is selected as the seed set of the positive cascade. For \( 1 \leq i \leq k \), let \( S_i = \arg \max_{|S| \leq i} f(S) \) and \( OPT_i = f(S_i) \). The rumor blocking problem considered in this paper is shown as follows.

**Problem 1.** Find a seed set \( S_p \) with at most \( k \) nodes such that \( f(S_p) \) is maximized.

It is well-known that this problem is NP-hard.

**Theorem 1.** [5] Problem is NP-hard.

### D. Realization

In this section, we introduce the concept of realization which provides the fundamental understanding of the IC model.

---

\( ^3 \)Due to space limitation, we skip the proof here. The interested reader is referred to [4] and [5].
Algorithm 1 Random R-tuple of \( (\mathcal{G}, S_v, v) \)

1. **Input:** the reverse network \( \mathcal{G} \), \( S_v \) and \( v \);
2. \( T_v(V) \leftarrow \{v\} \), \( T_v(E_i) \leftarrow \emptyset \), \( T_v(E_f) \leftarrow \emptyset \);
3. \( V' \leftarrow \emptyset \);
4. **while** \( V' \cap S_v = \emptyset \) **do**
5. \( T_v(V) \leftarrow T_v(V) \cup V' \);
6. \( V' \leftarrow \text{out}_{T_v(V)} \setminus T_v(V) \);
7. **for** each pair \( u_1 \) and \( u_2 \) where \( u_1 \in T_v(V) \) and \( u_2 \in V' \) **do**
8. \( \text{rand} \leftarrow \) a random number from 0 to 1 generated in uniform;
9. **if** \( \text{rand} \leq p^{G}_{(u_1,u_2)} \) **then**
10. \( V' \leftarrow V' \setminus \{u_2\} \);
11. \( T_v(E_i) \leftarrow T_v(E_i) \cup \{(u_2, u_1)\} \);
12. **else**
13. \( T_v(E_i) \leftarrow T_v(E_i) \cup \{(u_2, u_1)\} \);
14. Return \( (T_v(V), T_v(E_i), T_v(E_f)) \);

and

\[
\Pr[g] = \Pr[(E_1, E_2)] \cdot \Pr[g| (E_1, E_2)].
\]  

(3)

Intuitively, if a realization \( g \) is compatible to a valid pair \( (E_1, E_2) \), \( (E_1, E_2) \) can be taken as an intermediate state while generating \( g \). The concepts of realization and valid pair are important for the analysis in the next section.

**IV. THE ALGORITHM**

Before present the algorithm, we introduce several important terms.

**Definition 3. (Reverse network)** Given a network \( G \), the reverse network \( \mathcal{G} \) of \( G \) is constructed as follows. \( \mathcal{G} \) is identical to \( G \) except that \( p^{G}_{(u,v)} = p^{G}_{(v,u)} \). In brief, \( \mathcal{G} \) is obtained from \( G \) by reversing each edge in \( G \).

**Definition 4. (Random R-tuple of \( v \))** Generated by Algorithm 1 a random reverse tuple \( \mathcal{T}_v = (T_v(V), T_v(E_i), T_v(E_f)) \) of node \( v \) is a three tuple where \( T_v(V) \) is a node-set, and, \( T_v(E_i) \) and \( T_v(E_f) \) are edge-sets. Informally speaking, in the reserve graph \( \mathcal{G} \), we start from \( v \) and successively test whether the current neighbor of the nodes in \( T_v(V) \) can be added to \( T_v(V) \) in a breadth first manner until one of the rumor seed is reached. \( T_v(E_i) \) and \( T_v(E_f) \) are the generated in line 11 and line 13, respectively. We use \( T_v = (T_v(V), T_v(E_i), T_v(E_f)) \) to denote a concrete reverse tuple of \( v \). By abusing the notation, we also let \( T_v = \{T_v^1, T_v^2, \ldots \} \) be the set of all possible \( T_v \) and \( \Pr[T_v] \) be the probability that \( T_v \) can be generated by Algorithm 1.

\[
\Pr[T_v] = \prod_{e \in T_v(E_f)} p_{e}^{G} \prod_{e \in T_v(E_i)} (1 - p_{e}^{G}) = \Pr[(T_v^1(E_i), T_v^1(E_f))] \]  

(4)

Note that \( (T_v^1(E_i), T_v^1(E_f)) \) is always valid for each \( T_v^\ast \in \mathcal{T}_v \).

**Lemma 1.** For each node \( v \), all the \( C(T_v^i(E_i), T_v^i(E_f)) \) form a partition of \( \mathcal{G} \).

**Algorithm 2 Random R-tuple (\( \mathcal{G}, S_v \))**

1. **Input:** the reverse network \( \mathcal{G}, S_v \);
2. Randomly select a node \( v \) from \( V \) in uniform;
3. \( (T(V), T(E_i), T(E_f)) \leftarrow \text{Algorithm 1}(\mathcal{G}, S_v, v) \);
4. Return \( (T(V), T(E_i), T(E_f)) \)

**Proof.** First, it is obvious that for each \( g \in \mathcal{G} \) there exist a \( T_v^i \) such that \( g \) is compatible to \( (T_v^i(E_i), T_v^i(E_f)) \). Thus, it suffices to show that \( C(T_v^i(E_i), T_v^i(E_f)) \cap C(T_v^j(E_i), T_v^j(E_f)) = \emptyset \) for \( i \neq j \). Since \( T_v^i \) are different, there must be an edge \( e \) such that \( e \in T_v^i(E_i) \), \( e \notin T_v^i(E_f) \), \( e \notin T_v^j(E_i) \) and \( e \in T_v^j(E_f) \). By Def. 2 a realization \( g \) cannot be compatible to both \( T_v^i \) and \( T_v^j \). Lemma 1 thus proved.

**Definition 5. (Random R-tuple) Generated by Algorithm 2 a random R-tuple \( T = (T(V), T(E_i), T(E_f)) \) is a R-tuple \( T_v \) where \( v \) is selected from \( V \) uniformly in random.

**Definition 6.** Given a node set \( S \), let \( x(S, T) \) (resp. \( x(S, T_v) \)) be a random variable over \( 0 \) and \( 1 \), and \( x(S, T(V)) = 1 \) (resp. \( x(S, T_v(V)) = 1 \)) if and only if \( S \cap T(V) \neq \emptyset \) (resp. \( S \cap T_v(V) \neq \emptyset \)).

The following lemma is critical for the rest of the analysis in this section.

**Lemma 2.** \( E[x(S, T)] = \frac{\ell(S)}{n} \) where \( S \) is any subset of \( V \) and \( T \) is a random R-tuple.

**Proof.**

\[
E[x(S, T)] = \sum_{v} \sum_{T_v \in \mathcal{T}_v} \Pr[T_v] \cdot x(S, T_v)
\]

\[
= \sum_{v} \sum_{T_v \in \mathcal{T}_v} \Pr[T_v] \cdot \frac{\Pr[(T_v(E_i), T_v(E_f))] \cdot x(S, T_v)}{n}
\]

\[
= \sum_{v} \sum_{T_v \in \mathcal{T}_v} \Pr[(T_v(E_i), T_v(E_f))] \cdot \frac{\Pr[g|(T_v(E_i), T_v(E_f))] \cdot x(S, T_v)}{n}
\]

Note that for a realization \( g \) compatible to \( (T_v(E_i), T_v(E_f)) \) and a node set \( S \), by Property 1 \( v \) is not activated by rumor.
Algorithm 3 Node-Selection $(V, R_l, k)$

1: **Input:** $V, R_l = \{T^1, ..., T^l\}$ and $k$
2: $S' \leftarrow \emptyset$
3: for $j = 1 : k$ do
4: Let $v$ be the node that covers the most number of sets in $T^j(V)$;
5: $S' \leftarrow S' \cup \{v\}$;
6: Remove $v$ from each $T^j(V)$;
7: Return $S'$;

in $g$ under $S$ if and only if $x(S, T_v(V)) = 1$. Therefore,

$$
\sum_v \sum_{T_v \in T} \sum_{g \in C(T_v(E_1), T_v(E_2))} \Pr[g] \cdot x(S, T_v) = \sum_v \sum_{T_v \in T} \sum_{g \in C(T_v(E_1), T_v(E_2))} \Pr[g] \cdot f_S(T_v, v)
$$

{By Lemma\[1\]}

$$
\frac{n}{f(S)} \Pr[X \in \{S, R_l\}]
$$

Then

$$
F(S', R_l) \geq (1 - \frac{1}{e}) \cdot F(S, R_l),
$$

for any $S$ and $R_l$.

Let $X_i$ be $l$ i.i.d random variables where $E(X_i) = \mu$. The Chernoff bounds\[20\] states that

$$
\Pr\left[ \sum X_i - l \cdot \mu \geq \delta \cdot l \cdot \mu \right] \leq \exp\left(-\frac{l \cdot \mu \cdot \delta^2}{2 + \delta}\right),
$$

and

$$
\Pr\left[ \sum X_i - l \cdot \mu \leq -\delta \cdot l \cdot \mu \right] \leq \exp\left(-\frac{l \cdot \mu \cdot \delta^2}{2}\right),
$$

for $0 < \delta < 1$.

**Lemma 3.** Let $\delta_1 > 0$ and $N_1 > 0$ be the adjustable parameters. Pr$\left[ \frac{l}{n} \cdot F(S_k, R_l) \leq (1 - \delta_1) \cdot OPT_k \right] \leq 1/N_1$ if $l \geq l_1$ where

$$
l_1 = \frac{\lambda_1}{OPT_k},
$$

and

$$
\lambda_1 = \frac{2 \cdot n \cdot \ln N_1}{\delta_1^2}.
$$

**Proof.** Recall that $E[x(S_k, T)] = OPT_k / n$.

$$
\Pr\left[ \frac{l}{n} \cdot F(S_k, R_l) \leq (1 - \delta_1) \cdot OPT_k \right] = \Pr[F(S_k, R_l) \leq \frac{l}{n} \cdot (1 - \delta_1) \cdot OPT_k]
$$

$$
= \Pr[F(S_k, R_l) - \frac{l}{n} \cdot OPT_k \leq -\frac{l}{n} \cdot \delta_1 \cdot OPT_k]
$$

{By Eq.\[7\]}

$$
\leq \exp\left(-\frac{l \cdot OPT_k \cdot \delta_1}{2\delta_1^2}\right) \leq \frac{1}{N_1}
$$

The following result directly follows from the above lemma.

**Corollary 1.** $\frac{l}{n} \cdot F(S_k, R_l) \geq (1 - \delta_1) \cdot OPT_k$ holds with a probability larger than $1 - 1/N_1$ provided $l \geq l_1$.

**Lemma 4.** Let $\delta_1 > 0$, $\delta_2 > 0$ and $N_2 > 0$ be adjustable parameters.

$$
F(S, R_l) - \frac{l}{n} \cdot f(S) \leq (\delta_2 - (1 - e^{-1}) \cdot \delta_1) \cdot \frac{l}{n} \cdot OPT_k
$$

(10)

holds simultaneously for all $S \subseteq V$ with $|S| = k$ with a probability larger than $1 - 1/N_2$ if $l \geq l_2$ where

$$
l_2 = \frac{\lambda_2}{OPT_k},
$$

and

$$
\lambda_2 = \frac{(2 + \delta_2 - (1 - e^{-1}) \cdot \delta_1) \cdot n \cdot \ln(N_2 \cdot \binom{n}{k} \cdot \delta_1^2)}{(\delta_2 - (1 - e^{-1}) \cdot \delta_1)^2}
$$

(12)

**Proof.** Let $\delta_2 = (1 - e^{-1}) \cdot \delta_1$. For any fixed $S \subseteq V$ where $|S| = k$, it follows that

$$
\Pr[F(S, R_l) - \frac{l}{n} \cdot f(S) \geq (\delta_2 - (1 - e^{-1}) \cdot \delta_1) \cdot \frac{l}{n} \cdot OPT_k]
$$

{By Eq.\[9\]}

$$
\leq \exp\left(-\frac{l \cdot (\delta_2 - (1 - e^{-1}) \cdot \delta_1) \cdot f(S)}{n \cdot (2 + \delta_2 \cdot OPT_k)}\right)
$$

$$
\leq \exp\left(-\frac{l \cdot \delta_2 \cdot OPT_k}{n \cdot (2 + \delta_2)}\right)
$$

$$
= \frac{1}{N_2 \cdot \binom{n}{k}}
$$

Since there are at most $\binom{n}{k}$ subsets of $V$ with $k$ elements, by the union bound, Eq.\[10\] holds for all those sets with a probability larger than $1 - 1/N_2$.

Let $\lambda^* = \max(\lambda_1, \lambda_2)$. Now let us consider Algorithm 4. This algorithm simply generates $l^* = \frac{\lambda_1}{OPT_k}$ random R-tuples and then uses Algorithm 3 to find a seed set $S'$. The following result shows that the $S$ produced by Algorithm 4 is a good approximation with a high probability as long as $l^*$ is sufficiently large.
Theorem 2. Let $S'$ be the node-set produced by Algorithm 4. Then $f(S') \geq (1 - \varepsilon - 1 - \delta_2) \cdot OPT_k$ holds with a probability larger than $1 - 1/N_1 - 1/N_2$.

Proof. Suppose $\frac{n}{T} \cdot F(S_k, R) \geq (1 - \delta_1) \cdot OPT_k$ and Eq. [10] hold, then

$$f(S') \geq \frac{n}{T} \cdot F(S', T) - \delta \cdot OPT_k$$

{\textit{By Eq. [5]}}

$$\geq \frac{n}{T} \cdot (1 - \varepsilon - 1 - \delta_2) \cdot F(S_k, T) - \delta \cdot OPT_k$$

{\textit{By Corollary [4]}}

$$\geq (1 - \delta_1) \cdot (1 - \varepsilon - 1 - \delta_2) \cdot OPT_k$$

$$= (1 - \varepsilon - 1 - \delta_2) \cdot OPT_k$$

By Lemmas [3] and [4], Corollary [4] and Eq. [10], $f(S') \geq (1 - \varepsilon - 1 - \delta_2) \cdot OPT_k$ holds with a probability larger than $1 - 1/N_1 - 1/N_2$. $\square$

Running time. Let $TIME$ be the expected running time of Algorithm 4. The following lemma shows $TIME$ can be bounded by the objective value of the optimal solution.

Lemma 5. $TIME \leq \frac{m}{n} \cdot OPT_1$

Proof. For a R-tuple $T_v$ of node $v$, let $TIME(T_v)$ be the time consumed in generating $T_v$. Thus,

$$TIME = \frac{\sum_v \sum_{T_v} \Pr[T_v] \cdot TIME(T_v)}{n}$$

Note that $TIME(T_v)$ is equal to the number of edges tested during the generation of $T_v$. Thus,

$$\sum_v \sum_{T_v} \Pr[T_v] \cdot TIME(T_v)$$

$$= \sum_v \sum_{T_v} \Pr[T_v] \cdot \sum_{(u^*, v^*) \in E} x(\{u^*\}, T_v)$$

$$= \sum_{(u^*, v^*) \in E} \sum_v \sum_{T_v} \Pr[T_v] \cdot x(\{u^*\}, T_v)$$

$$= \sum_{(u^*, v^*) \in E} f(\{u^*\})$$

$$\leq \frac{m \cdot OPT_1}{n}$$

$\square$

Corollary 2. If $l^* = O(\frac{\lambda}{OPT_k})$, then the running time of Algorithm 4 is $O(k \cdot m \cdot \ln n)$.

Now we can see that Algorithm 4 is an effective and efficient rumor blocking algorithm. However, $OPT_k$ is unknown to us.

Algorithm 4 Positive-Seed-Detection ($G, \lambda^*, OPT_k, S_r, k$)

1: Input: $G, \lambda^*, OPT_k, S_r$ and $k$
2: Generate $l^* = \frac{\lambda^*}{OPT_k}$ random reverse sets $R_i = \{T^1, ..., T^k\}$
3: Run Algorithm 2 with input $R_i$ and $k$ to obtain a node set $S'$
4: Return $S'$

Algorithm 5 OPT$_k$-Estimation ($G, k, \delta_3$)

1: Input: $G = (V, E)$ and $k$
2: $R \leftarrow \emptyset$
3: $\lambda_3 = \frac{n \cdot (2 + \delta_3) \ln(N_3 \cdot (k) \log n)}{\delta_3^2}$
4: for $i = 1: \log(n - 1)$ do
5: $x_i \leftarrow \frac{2}{n}$, $l_i \leftarrow \frac{x_i}{x_r^2}$
6: while $|R| \leq l_i$ do
7: Generate a random R-tuple and insert it into $R$
8: $S' \leftarrow$ Node-Selection($V, R, k$)
9: if $\frac{n \cdot F(S'_R, R)}{l_i} \geq (1 + \delta_3) \cdot x_i$ then
10: $OPT_k = \frac{n \cdot F(S'_R, R)}{l_i}$
11: Return $OPT_k$

We will in the following show that $OPT_k$ can be estimated by sampling R-tuples. It is desired to find an estimate $\overline{OPT}_k$ of $OPT_k$ such that $\overline{OPT}_k$ is less than $OPT_k$ but not very far from $OPT_k$. Intuitively, $\frac{n \cdot F(S'_R, R)}{l_i}$ should be a good choice as it is close to $\frac{n \cdot F(S'_R, R)}{l_i}$ with a guaranteed factor and $\frac{n \cdot F(S'_R, R)}{l_i}$ is an unbiased estimate of $OPT_k$. Note that $1 \leq OPT_k$ $\leq n$. Therefore, we design a statistic test which compares $\frac{n \cdot F(S'_R, R)}{l_i}$ with $n/2^*$. We choose the probability $\delta_3 > 0$.

First, we need to guarantee that $\overline{OPT}_k$ is smaller than $OPT_k$. The following result shows that the terminate condition (i.e., line 9) leads that $\overline{OPT}_k$ is smaller than $OPT_k$ with a high probability.

Lemma 6. For the $i$-th iteration from line 6 to 11 in Algorithm 5, the terminate condition holds with at most $\frac{1}{N_3 \cdot \log n}$ probability provided $OPT_k < x_c$.

Proof. For a node-set $S$ with $|S| = k$,

$$\Pr[n \cdot F(S, R) / l_i \geq (1 + \delta_3) \cdot x_i]$$

$$= \Pr[F(S, R) / l_i \geq \frac{l_i \cdot f(S)}{n} \geq \frac{l_i}{n} \cdot f(S)]$$

{\textit{By Eq. [5]}}

$$\leq \exp(-\frac{-l_i \cdot f(S)}{2 \cdot \frac{(1 + \delta_3) \cdot x_i}{f(S)} - 1})$$

$$\leq \exp(-\frac{-l_i \cdot f(S)}{2 \cdot \frac{(1 + \delta_3) \cdot x_i}{f(S)} + 1})$$

{\textit{Since $f(S) \leq OPT < x_c$}}

$$\leq \exp(-\frac{-l_i \cdot f(S)}{2 \cdot \frac{(1 + \delta_3) \cdot x_i}{f(S)} + 1})$$

$$= \exp(\frac{-l_i \cdot f(S)}{n \cdot (2 + \delta_3) \cdot \log n} \cdot \frac{1}{N_3 \cdot (k) \log n})$$

By the union bound, $n \cdot F(S, T) / l_i \geq (1 + \delta_3) \cdot x_i$ holds for
the $S$ produced in line 8 Algorithm 5 with a probability less than $\frac{1}{N_3 \log n}$.

**Corollary 3.** With at least $1 - \frac{1}{N_3}$ probability, Algorithm 5 will not terminate in the $i$-th round with $OPT_k < x_i$.

**Proof.** According to Lemma 3 the probability that Algorithm 5 terminates is at most $\frac{1}{N_3 \log n}$ when $i \leq \log i / OPT_k$. By the union bound, the probability that Algorithm 5 terminates before $i = \log n / OPT_k$ is less than $\frac{1}{N_3 \log n} \leq \frac{1}{N_3}$. Thus, proved.

**Lemma 7.** If $OPT_k \geq x_i$, then $OPT_k \leq OPT_k$ holds with a probability at least $1 - 1/N_3$.

**Proof.** For an node-set $S$ with $|S| = k$,

$$
\Pr \left[ \frac{n \cdot F(S, T)}{l_i} \geq OPT_k \right] = \Pr \left[ \frac{n \cdot F(S, T)}{l_i} \geq l_i \cdot (1 + \delta_3) \cdot OPT_k \right]
$$

$$
\leq \Pr \left[ F(S, T) \geq \frac{l_i \cdot f(S)}{n} \cdot OPT_k \cdot \delta_3 \right]
$$

$$
\leq \exp \left( - \frac{l_i \cdot f(S)}{2 + \frac{OPT_k \cdot \delta_3}{f(S)}} \right)
$$

$$
= \exp \left( - \frac{l_i \cdot OPT_k \cdot \delta_3}{n \cdot (2 \cdot f(S) + OPT_k \cdot \delta_3)} \right)
$$

$$
= \exp \left( - \frac{l_i \cdot OPT_k \cdot \delta_3}{n \cdot (2 + \delta_3)} \right)
$$

$$
= \exp \left( - \frac{OPT_k}{x} \cdot \ln \left( \frac{n}{k} \cdot N_3 \right) \right)
$$

$$
\leq 1 / \left( \log n \cdot \frac{n}{k} \cdot N_3 \right)
$$

The lemma thus follows by the union bound.

The following result directly follows from Corollary 3 and 7.

**Theorem 3.** With probability less than $1 - 2/N_3$, Algorithm 5 produces an $OPT_k$ that is less $OPT_k$.

Second, it can be shown that $OPT_k$ is not too much less than $OPT_k$, which guarantees that $l^* = O \left( \frac{BRB^*}{OPT_k} \right)$.

**Lemma 8.** If $OPT_k \geq \frac{1 + \delta_3}{x_i}$, then $n \cdot F(S', T) / l_t \leq (1 + \delta_3) \cdot x_i$ holds with at most $1/N_3$ probability.

**Algorithm 6 R-tuple Based Randomized Rumor Blocking**

1. **Input:** $G$, $S_r$, $k$, $N_1$, $N_2$, $N_3$, $\delta_1$, $\delta_2$ and $\delta_3$.
2. $\lambda_1 = \frac{2 \ln N_1}{l_t}$.
3. $\lambda_2 = \frac{(2 + \delta_2) - (1 - e^{-1}) \delta_3}{(2 - \delta_1) \delta_3}$.
4. $\lambda_3 = \frac{\delta_3}{\delta_3}$.
5. $\lambda^* = \max (\lambda_1, \lambda_2)$.
6. $OPT_k \rightarrow$ Algorithm 5 $(G, k, \delta_3)$.
7. $S' \rightarrow$ Algorithm 4 $(G, \lambda^*, OPT_k, S_r, k)$.
8. Return $S'$.

**Proof.**

$$
\Pr \left[ n \cdot F(S', T) / l_i \leq (1 + \delta_3) \cdot x_i \right]
\leq \Pr \left[ n \cdot (1 - e^{-1}) F(S, T) / l_i \leq (1 + \delta_3) x_i \right]
\leq \Pr \left[ n \cdot F(S, T) / l_i \leq \frac{OPT_k}{1 + \delta_3} \right]
$$

$$
= \Pr \left[ F(S, T) \leq \frac{l_i \cdot f(S)}{n} \cdot \frac{OPT_k}{1 + \delta_3} \right]
$$

$$
\leq \exp \left( - \frac{l_i \cdot f(S)}{n} \cdot \frac{OPT_k}{1 + \delta_3} \right)
$$

$$
\leq \exp \left( - \frac{l_i \cdot OPT_k \cdot \delta_3}{n \cdot (1 + \delta_3)} \right)
$$

$$
\leq 1 / \left( \frac{n}{k} \cdot N_3 \right)
$$

By the union bound, the above holds for the $S'$ produced in line 8 Algorithm 5 with a probability less than $1/N_3$.

**Theorem 4.** With a probability at least $1 - 1/N_3$, Algorithm 5 produces an $OPT_k$ such that $OPT_k \geq \frac{(1 - e^{-1}) \cdot OPT_k}{2(1 + \delta_3)^2}$.

**Proof.** Suppose $n / 2^{i+1} \leq \frac{OPT_k}{(1 + \delta_3)^2} \leq n / 2^i$ for some $i$. If Algorithm 5 terminates before the $(i + 1)$-th iteration, then, by line 9 in Algorithm 5, $OPT_k \geq n / 2^i \geq \frac{OPT_k}{(1 + \delta_3)^2}$. Now suppose Algorithm 5 enters the $(i + 1)$-th iteration. By Lemma 8, it will terminate with a probability larger that $1 - 1/N_3$, which means $OPT_k \geq n / 2^{i+1} \geq \frac{OPT_k}{2(1 + \delta_3)^2}$.

The above results are summarized as follows.

**Theorem 5.** With a probability at least $1 - 3/N_3$, Algorithm 5 returns an $OPT_k$, such that

$$
OPT_k \geq OPT_k \geq \frac{(1 - e^{-1}) \cdot OPT_k}{2(1 + \delta_3)^2} \quad (13)
$$

The RBR algorithm. According to the analysis above, our randomized rumor blocking framework is shown in Algorithm 6, denoted as reverse-tuple based randomized (RBR) algorithm. We first apply Algorithm 5 to obtain an estimate $OPT_k$ of $OPT_k$ and then find a seed set $S'$ using Algorithm 4 with the input $(G, \lambda^*, OPT_k, S_r, k)$. By Theorems 3 and 7 and Corollary 2, $F(S') \geq (1 - e^{-1} - \delta_2) \cdot OPT_k$ holds with a high probability and the expected running time of
Algorithm [6] is $O\left(\frac{m \ln n}{\delta^2}\right)$. In the experiment, we simply set that $N_1 = N_2 = N_3 = n$, $\delta_2 = \delta_3 = 0.1$. Now the only undetermined parameter is $\delta_1$. According to Algorithm [4] we select the $\delta_1$ such that $\lambda^*$ can be minimized.

V. EXPERIMENT

In this section, we evaluate the performance of the RBR algorithm with respect to the state-of-the-art method and other heuristics. Besides, we also discuss the running time of the considered algorithms.

In our experiments, we employ three datasets, Power2500, Wiki and Epinion, scaling from small to large. Power2500 is a synthetic power-law network generated by DIGG [21]. It has been shown that the power-law distribution is one of the most important characteristics of social networks [22]. Wiki is a who-votes-on-whom network extracted from the vote history data of Wikipedia. Epinions is a who-trust-whom online social network extracted from the consumer review site Epinions.com. Wiki and Epinions are provided by the SNAP [7].

The basic statistics of the above datasets are shown in Table I.

![Fig. 1: Running time on Power2500 under the log-normal scale.](image)

TABLE I: Datasets

| Dataset  | Node | Edge | Average out-Degree |
|----------|------|------|--------------------|
| Power2500 | 2.5K | 26K  | 20.8               |
| Wiki     | 7K   | 30K  | 12.0               |
| Epinions | 75K  | 508K | 13.4               |

The probability on the edges is either uniformly set as 0.1 or $p_{(u,v)}^G$ is set as $1/\text{degree}(v)$ where $\text{degree}(v)$ is the in-degree of node $v$. The above datasets together with the probability settings are widely used in the prior works.

We consider four rumor blocking algorithms shown as follows:

- **RBR algorithm.** This is the randomized algorithm proposed in this paper. As discussed in Sec. [IV] we set the error probability as $O(1/n)$.
- **Greedy.** This is the state-of-the-art rumor blocking algorithm using the Monte-Carlo simulation. Following the conventions, 10,000 simulations are run for each estimation.
- **Proximity.** This is a popular heuristic algorithm which selects the out-neighbors of the rumor seed nodes as the positive seed nodes. In particular, we give an index to each node and select the neighbors with the highest index.
- **Random.** This is a baseline method where the positive seed nodes are randomly selected.
- **Unblocking.** This is the base case when there is no positive cascade.

In our experiments, the rumor seed nodes are selected from the nodes with the highest degree. For the three datasets, the numbers of seed nodes are set as 10, 10 and 20, respectively, and the budgets of the positive cascades are set as 20, 30 and 50, respectively. As mentioned in Sec. [IV] the parameter $\delta_1$ is selected to minimize $\lambda^*$ according to Eqs. [9] and [12].

A. Results

For the Power2500 network, the results under the two settings of the propagation probability are shown in Figs. 2a and 2b, respectively. We can see that the RBR algorithm achieves more than 95% blocking effect of the state-of-the-art algorithm. For example, in Fig. 2b when $k = 10$, the numbers of rumor-activated nodes under RBR algorithm and Greedy algorithm are 560 and 550 respectively. Nevertheless, the RBR algorithm is more efficient than the Greedy algorithm with respect to the running time. The running time (under the log-normal scale) of the two algorithms on Power2500 are shown in Figs. 1a and 1b. We can see that the running time of RBR algorithm is much less than that of the Greedy algorithm. For example, when the propagation probability is 0.1, Greedy algorithm requires 24.13 hours to select 15 positive seed node while RBR algorithm only takes 17 minutes.

The results on the Wiki dataset is shown in Figs. 2c and 2e. On this network, the RBR algorithm is able to reach at least 97.98% blocking effect of the Greedy algorithm. Despite that Wiki is larger than Power2500, comparing Figs. 2a and 2c one can see that when there is no positive cascade, 10 rumor seed nodes result 285 and 460 rumor-activated nodes on Wiki and Power2500, respectively. Such an observation indicates that the dense of the network has more effect on the influence diffusion than the network scale does.

The results on the Epinions dataset are shown in Figs. 2d and 2f where the Greedy algorithm is excluded as it is extremely time consuming. One can see that on the large network the RBR algorithm is superior to other heuristics by a significant margin. Under the setting that $p_{(u,v)}^G = 0.1$ and $k = 50$, the RBR algorithm can protect about 2000 users while Proximity protects 800 nodes and Random can hardly protect any node.

VI. CONCLUSION

In this paper, we have studied the rumor blocking problem for online social networks. We first design the reverse-tuple based sample method and then present a randomized rumor blocking algorithm which guarantees a $(1 - 1/e - \delta)$-approximation and runs in time $O\left(\frac{km \ln n}{\delta^2}\right)$ with a high probability. The proposed RBR algorithm theoretically dominates the existing rumor blocking algorithms, and as shown in the...
experiments it is very efficient without sacrificing the blocking effect.

ACKNOWLEDGMENT

This work was supported by National Natural Science Foundation of China under 61472272.

REFERENCES

[1] E. Morozov, “Swine flu: Twitter’s power to misinform,” Foreign policy, 2009.
[2] D. Kempe, J. Kleinberg, and É. Tardos, “Maximizing the spread of influence through a social network,” in Proc. of the 9th SIGKDD. ACM, 2003, pp. 137–146.
[3] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, “An analysis of approximations for maximizing submodular set functions,” Mathematical Programming, vol. 14, no. 1, pp. 265–294, 1978.
[4] X. He, G. Song, W. Chen, and Q. Jiang, “Influence blocking maximization in social networks under the competitive linear threshold model,” in SDM. SIAM, 2012, pp. 463–474.
[5] C. Budak, D. Agrawal, and A. El Abbadi, “Limiting the spread of misinformation in social networks,” in Proc. of the 20th WWW. ACM, 2011, pp. 665–674.
[6] L. Fan, Z. Lu, W. Wu, B. Thuraisingham, H. Ma, and Y. Bi, “Least cost rumor blocking in social networks,” in Proc. of the 33rd ICDCS. IEEE, 2013, pp. 540–549.
[7] W. Chen, C. Wang, and Y. Wang, “Scalable influence maximization for prevalent viral marketing in large-scale social networks,” in Proc. of the 16th SIGKDD. ACM, 2010, pp. 1029–1038.
[8] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier, “Maximizing social influence in nearly optimal time,” in Proc. of the 25th SODA. SIAM, 2014, pp. 946–957.
[9] Y. Tang, X. Xiao, and Y. Shi, “Influence maximization: Near-optimal time complexity meets practical efficiency,” in Proc. of SIGMOD. ACM, 2014, pp. 75–86.
[10] Y. Tang, Y. Shi, and X. Xiao, “Influence maximization in near-linear time: a martingale approach,” in Proc. of SIGMOD. ACM, 2015, pp. 1539–1554.
[11] J. Leskovec, L. Backstrom, and J. Kleinberg, “Meme-tracking and the dynamics of the news cycle,” in Proc. of the 15th SIGKDD. ACM, 2009, pp. 497–506.
[12] J. Ratkiewicz, M. Conover, M. Meiss, B. Gonçalves, A. Flammini, and F. Menczer, “Detecting and tracking political abuse in social media,” in Proc. of the 5th ICWSM, 2011.
[13] V. Qazvinian, E. Rosengren, D. R. Radev, and Q. Mei, “Rumor has it: Identifying misinformation in microblogs,” in Proc. of EMNLP. Association for Computational Linguistics, 2011, pp. 1589–1599.
[14] T. Takahashi and N. Igata, “Rumor detection on twitter,” in Proc. of the 6th ICIS-ISIS.
[15] D. Shah and T. Zaman, “Rumors in a network: who’s the culprit?” IEEE Transactions on Information Theory, vol. 57, no. 8, pp. 5163–5181, 2011.
[16] Z. Wang, W. Dong, W. Zhang, and C. W. Tan, “Rumor source detection with multiple observations: Fundamental limits and algorithms,” in Proc. of IGMTRICS, vol. 42, no. 1. ACM, 2014, pp. 1–13.
[17] S. Li, Y. Zhu, D. Li, D. Kim, and H. Huang, “Rumor restriction in online social networks,” in Proc. of the 32nd IPCCC. IEEE, 2013, pp. 1–10.
[18] H. Zhang, H. Zhang, X. Li, and M. T. Thai, “Limiting the spread of misinformation while effectively raising awareness in social networks,” in Proc. of the International Conference on Computational Social Networks. Springer, 2015, pp. 35–47.
[19] Y. Ping, Z. Cao, and H. Zhu, “Sybil-aware least cost rumor blocking in social networks,” in Proc. of the 2014 IEEE Global Communications Conference. IEEE, 2014, pp. 692–697.
[20] R. Motwani and P. Raghavan, Randomized algorithms. Chapman & Hall/CRC, 2010.
[21] L. Cowen, A. Brady, and P. Schmid, “Digg: Dynamic graph generator.”
[22] A. Clauset, C. R. Shalizi, and M. E. Newman, “Power-law distributions in empirical data,” SIAM review, vol. 51, no. 4, pp. 661–703, 2009.