Scalable Misinformation Mitigation in Social Networks using Reverse Sampling

Michael Simpson[1], Venkatesh Srinivasan[2] and Alex Thomo[2]

[1] *University of British Columbia, Vancouver, Canada*
[2] *University of Victoria, Victoria, Canada*
Email: mesimp@cs.ubc.ca, {srinivas,thomo}@uvic.ca

We consider misinformation propagating through a social network and study the problem of its prevention. The goal is to identify a set of k users that need to be convinced to adopt a limiting campaign so as to minimize the number of people that end up adopting the misinformation. This work presents *RPS* (Reverse Prevention Sampling), an algorithm that provides a scalable solution to the misinformation mitigation problem. Our theoretical analysis shows that *RPS* runs in $O((k+l)(n+m)(\frac{1}{1-\gamma})\log n/\epsilon^2)$ expected time and returns a $(1-1/e-\epsilon)$-approximate solution with at least $1-n^{-l}$ probability (where γ is a typically small network parameter and l is a confidence parameter). The time complexity of *RPS* substantially improves upon the previously best-known algorithms that run in time $\Omega(mnk \cdot POLY(\epsilon^{-1}))$. We experimentally evaluate *RPS* on large datasets and show that it outperforms the state-of-the-art solution by several orders of magnitude in terms of running time. This demonstrates that misinformation mitigation can be made practical while still offering strong theoretical guarantees.

Keywords: Graph Algorithms, Social Networks, Misinformation Prevention

1. INTRODUCTION

Social networks allow for the widespread distribution of knowledge and information in modern society as they have rapidly become a place to hear the news and discuss social topics. Information can spread quickly through the network, eventually reaching a large audience, especially so for influential users. While the ease of information propagation in social networks can be beneficial, it can also have disruptive effects. In recent years, the number of high profile instances of misinformation causing severe real-world effects has risen sharply. These examples range across a number of social media platforms and topics [1, 2, 3, 4, 5, 6]. For example, a series of bogus tweets from a trusted news network referring to explosions at the White House caused immediate and extensive repercussions in the financial markets [1]. During a recent shooting at YouTube's headquarters, and before police managed to secure the area, a wave of misinformation and erroneous accusations were widely disseminated on Twitter causing panic and confusion [2, 3]. Finally, there has been much discussion on the role misinformation and fake news played in the 2016 U.S. presidential election with sites such as Reddit and Facebook being accused of harbouring and spreading divisive content and misinformation [4, 5, 6]. Thus, in order for social networks to serve as a reliable platform for disseminating critical information, it is necessary to have tools to limit the spread of misinformation.

Budak et al. [7] were among the first to formulate the problem of misinformation mitigation as a combinatorial optimization problem. By building upon the seminal work of Kempe et al. [8] on *influence maximization*, they introduce two multi-campaign propagation models, the Multi-Cascade Independent Cascade (MCIC) and Campaign-Oblivious Independent Cascade (COIC) models, and present a greedy approach that provides a $(1-1/e-\epsilon)$-approximate solution. Unfortunately, their greedy approach is plagued by the same scaling issues as [8] when considering large social networks and is further exacerbated by the added complexity of tracking multiple cascades which requires costly shortest path computations. This leads us to the motivating question for this paper: Can we find scalable algorithms for the misinformation mitigation problem introduced in [7] under the MCIC model?

The scalability hurdle in the *single* campaign setting was resolved by Borgs et al. [9] when they made a theoretical breakthrough that fundamentally shifts the way in which we view the influence maximization problem. Their key insight was to reverse the question of "what subset of the network can a particular user

influence" to "who could have influenced a particular user". Their sampling method runs in close to linear time and returns a $(1 - 1/e - \epsilon)$-approximate solution with at least $1 - n^{-l}$ probability. In subsequent work, Tang et al. [10, 11] presented a significant advance that improved the practical efficiency through a careful theoretical analysis that rids the Borgs et al. approach of a large hidden constant in the runtime guarantee.

In this work, we achieve scalability for the misinformation mitigation problem in the MCIC model. We complement our theoretical analysis with extensive experiments which show an improvement of several orders of magnitude over Budak et al. [7]. Since influence in the single campaign setting corresponds to reachability in the network, our solution requires mapping the concept of reachability to an analogous notion in the multi-campaign model for misinformation mitigation. Our first contribution is to show that reachability alone is not sufficient in determining the ability to save a particular node from the bad campaign. In order to address this challenge, we introduce a crucial notion of "obstructed" nodes, which are nodes such that all paths leading to them can be blocked by the bad campaign.

Using our newly defined notion of obstruction, we develop an efficient algorithm for the misinformation mitigation problem that provides much improved scalability over the existing Monte Carlo-based greedy approach of [7]. A novel component of this algorithm is a procedure to compute the set of unobstructed nodes that could have saved a particular node from adopting the misinformation. We obtain theoretical guarantees on the expected runtime and solution quality for our new approach and show that its expected runtime substantially improves upon the expected runtime of [7]. Additionally, we rule out sublinear algorithms for our problem through a lower bound on the time required to obtain a constant approximation.

Finally, from an experimental point of view, we show that our algorithm gives a significant improvement over the state of the art algorithm and can efficiently handle graphs with more than 50 million edges. In summary, the contributions of this paper are:

1. We introduce the concept of *obstructed* nodes that fully captures the necessary conditions for preventing the adoption of misinformation in the multi-campaign model. In the process, we close a gap in the work of [7].
2. We design and implement a novel procedure for computing the set of nodes that could save a particular user from adopting the misinformation.
3. We propose a misinformation mitigation approach that returns a $(1 - 1/e - \epsilon)$-approximate solution with high probability in the multi-campaign model and show that its expected runtime substantially improves upon that of the algorithm of Budak et al. [7].
4. We give a lower bound of $\Omega(m + n)$ on the time required to obtain a constant approximation for the misinformation mitigation problem.
5. Our experiments show that our algorithm gives an improvement of several orders of magnitude over Budak et al. [7] and can handle graphs with more than 50 million edges.

2. RELATED WORK

Influence Maximization There exists a large body of work on the Influence Maximization problem first proposed by Kempe et al. [8]. The primary focus of the research community has been related to improving the practical efficiency of the Monte Carlo-based greedy approach under the Independent Cascade (IC) or Linear Threshold (LT) propagation models. These works fall into two categories: heuristics that trade efficiency for approximation guarantees [12, 13] and practical optimizations that speed up the Monte Carlo-based greedy approach while retaining the approximation guarantees [14, 15, 16]. Despite these advancements, it remains infeasible to scale the Monte Carlo-based approach to web-scale networks.

Borgs' et al. [9] brought the first asymptotic runtime improvements while maintaining the $(1 - 1/e - \epsilon)$-approximation guarantees with their *reverse influence sampling* technique. Furthermore, they prove their approach is near-optimal under the IC model. State-of-the-art solutions to the IM problem [10, 11, 17, 18, 19] rely on reverse sampling for their efficiency.

Incorporating the spread of multiple campaigns is split between two main lines of work: (1) studying influence maximization in the presence of competing campaigns [20, 21, 22, 23] and (2) limiting the spread of misinformation and rumours by launching a truth campaign [7, 24, 25, 26, 27, 28, 29, 30]. In both cases, existing propagation models (such as IC and LT) are augmented or extended.

Misinformation Mitigation Mitigation refers to how and by what means we can combat or prevent the spread of misinformation that is currently spreading through a network. The misinformation mitigation problem was first studied under an independent cascade model by Budak et al. [7] and under a linear threshold model by He et al. [24]. Unfortunately, despite the objective function proving to be monotone and submodular, the Monte Carlo-based greedy solutions used in [7, 24] face the same scalability challenges as [8]. The related problem of determining the budget required to ensure that a fixed fraction of the network remains free of misinformation was investigated in [25, 31, 32].

More recently [33, 29, 26, 27, 28, 30] extend the *reverse influence sampling* technique of [9] to spreading truth to combat misinformation. This line of work aims to incorporate ideas from the state-of-the-art reverse sampling techniques used for the IM problem to solving

the misinformation mitigation problem. However, their work differs from ours in an important way: they make use of the COIC model where the edge probabilities are *campaign oblivious*. This alternative model does not capture the notion of misinformation spread as well as the MCIC model due to the assumption that users adopt truth and misinformation with identical probability (see [7] for a discussion). Furthermore, the shared edge probability assumption comes with added theoretical benefits that greatly simplify the adaptation of state-of-the-art reverse sampling solutions for the IM problem to the misinformation mitigation problem. In this work, we make a first step towards incorporating these recent advances to the more challenging setting under the MCIC model.

Fake News Intervention The intervention problem seeks to take stronger actions on fake items including content removal, account suspension and tagging content with warning labels. These are often inspired by immunizations techniques for epidemics [34, 35, 36, 37] and involve actions such as edge manipulation [38, 39, 40], node removal [41] and immunizing vulnerable nodes [42]. Recently, there has been a concerted effort from major social media websites including Facebook, Twitter, Instagram, and Pinterest towards combating fake news [43, 44, 45, 46, 47, 48] that includes the use of warning labels on content that has been identified as false or misleading.

Fake News Detection The detection problem seeks to identify which items (such as post, tweets, articles) are fake. See [49] for a recent survey. This area has seen much attention recently by the database and machine learning communities among others. Detecting fake content was cast as a few-shot rare category learning problem in [50]. An approach for mining user-specified network structures was proposed in [51] which can aid in detecting botnets that are posting misinformation. In [52] an end-to-end fact checking system was proposed. The work of [53, 54, 55] aimed to leverage crowdsourcing approaches to improve fake news detection. In [56, 57, 58], a natural interpretation of fact checking to the classical database problems of data integration and truth discovery were studied. Recently, the proliferation of high-quality knowledge graphs has provided an opportunity to enhance fact checking capabilities by searching for supporting evidence in a knowledge graph [59, 60, 61, 62]. Finally, a collection of studies [63, 64, 65] investigated how to identify fake news by considering the credibility of the sources of social media content.

3. PRELIMINARIES

In this section, we formally define the multi-campaign diffusion model, the eventual influence limitation problem presented by Budak et al. [7], and present an overview of the state-of-the-art reverse sampling approach [8, 9, 10] for the influence maximization problem.

Diffusion Model Let C (for "bad *C*ampaign") and L (for "*L*imiting") denote two influence campaigns. Let $\mathcal{G} = (V, E, p)$ be a social network with node set V and directed edge set E ($|V| = n$ and $|E| = m$) where p is a function that specifies *campaign-specific* pairwise influence probabilities (or weights) between nodes. That is, $p : E \times Z \rightarrow [0, 1]$ where $Z \in \{C, L\}$. For convenience, we use $p_Z(e)$ to denote $p(e, Z)$. Further, let $G = (V, E)$ denote the underlying unweighted directed graph. Given $\mathcal{G}$, the Multi-Campaign Independent Cascade model (MCIC) of Budak et al. [7] considers a time-stamped influence propagation process as follows:

1. At timestamp 1, we *activate* selected sets A_C and A_L of nodes in $\mathcal{G}$ for campaigns C and L respectively, while setting all other nodes *inactive*.
2. If a node u is first activated at timestamp i in campaign C (or L), then for each directed edge e that points from u to an inactive neighbour v in C (or L), u has $p_C(e)$ (or $p_L(e)$) probability to activate v at timestamp $i + 1$. After timestamp $i + 1$, u cannot activate any node.
3. In the case when two or more nodes from different campaigns are trying to activate v at a given time step we assume that the "good information" (i.e. campaign L) takes effect.
4. Once a node becomes activated in one campaign, it never becomes inactive or changes campaigns.

He et. al. [24] consider the opposite policy to (3) where the misinformation succeeds in the case of a tie-break. We note that our algorithms presented in this work are applicable for both choices of the tie-break policy.

3.1. Formal Problem Statement

A natural objective, as outlined in [7], is "saving" as many nodes as possible. That is, we seek to minimize the number of nodes that end up adopting campaign C when the propagation process is complete. This is referred to as the *eventual influence limitation problem (EIL)*.

Let A_C and A_L be the set of nodes from which campaigns C and L start, respectively. Let $I(A_C)$ be the set of nodes that are activated in campaign C in the absence of L when the above propagation process converges and $\pi(A_L)$ be the size of the subset of $I(A_C)$ that campaign L prevents from adopting campaign C. We refer to A_L and A_C as the *seed sets*, $I(A_C)$ as the *influence* of campaign C, and $\pi(A_L)$ as the *prevention* of campaign L. The nodes that are prevented from adopting campaign C are referred to as *saved*. Note that $\pi(A_L)$ is a random variable that depends on the

edge probabilities that each node uses in determining out-neighbors to activate.

Budak et al. [7] present a simplified version of the problem that captures the idea that it may be much easier to convince a user of the truth. Specifically, the information from campaign L is accepted by users with probability 1 ($p_L(e) = 1$ if edge e exists and $p_L(e) = 0$ otherwise) referred to as the *high effectiveness property*. In [7] it is shown that even with these restrictions EIL with the high effectiveness property is NP-hard. Interestingly, with the high effectiveness property, the prevention function is submodular and thus a Monte Carlo-based greedy approach (referred to here as *MCGreedy*) yields approximation guarantees.

We motivate the high effectiveness property with the following two real-world scenarios: (1) the phenomenon of "death hoaxes" (where celebrities or other notable figures are claimed to have died) have a strong corrective measure when the victim, or a close relative, makes an announcement on their personal account that contradicts the rumour and (2) false reporting of natural disasters can be countered by trusted news organizations providing coverage of the location of the purported scene. In both cases, the sharing of links to strong video, photographic, or text evidence that is also coming from a credible source lends itself to a scenario following the high effectiveness property. In addition to the scenarios we have outlined, the model is attractive because this assumption leads to interesting theoretical guarantees. Budak et al. study and obtain results for EIL with the high effectiveness property and is the problem that we consider in this work.

Problem 1. Given $\mathcal{G}$, seed set A_C, and a positive integer k, the eventual influence limitation (EIL) problem asks for a size-k seed set A_L maximizing the value of $\mathbb{E}[\pi(A_L)]$ under the MCIC model with the high effectiveness property.

Possible Worlds Interpretation To facilitate a better understanding of MCIC, we define a *Possible World (PW) model* that provides an equivalent view of the MCIC model and follows a widely used convention when studying IM and related problems [8, 7, 15, 16, 21, 24, 7, 25, 66]. Given a graph $\mathcal{G} = (V, E, p)$ and the MCIC diffusion model, a possible world X consists of two *deterministic graphs*, one for each campaign, sampled from a probability distribution over $\mathcal{G}$. The stochastic diffusion process under the MCIC model has the following equivalent description: we can interpret $\mathcal{G}$ as a distribution over unweighted directed graphs, where each edge e is independently realized with probability $p_C(e)$ (or $p_L(e)$). Observe, given the high effectivness property, the deterministic graph that defines the possible world for campaign L is simply the underlying unweighted graph G. Then, if we realize a graph g according to the probability distribution given by $p_C(e)$, we can associate the set of saved nodes in the original process with the set of nodes which campaign L reaches before campaign C during a *deterministic* diffusion process in $g \sim \mathcal{G}$ by campaign C and in G by campaign L. That is, we can compute the set of saved nodes with a deterministic cascade in the resulting possible world $X = (g, G)$. The following theorem from [67] establishes the equivalence between this possible world model and MCIC. This alternative PW model formulation of the EIL problem under the MCIC model will be used throughout the paper.

Theorem 1 ([67]). For any fixed seed sets A_C and A_L, the joint distributions of the sets of C-activated nodes and L-activated nodes obtained (i) by running a MCIC diffusion from A_C and A_L and (ii) by randomly sampling a possible world $X = (g, G)$ and running a deterministic cascade from A_C in g and A_L in G, are the same.

3.2. Reverse Sampling for Influence Maximization

In this section we review the state-of-the-art approach to the well studied *influence maximization problem (IM)*. This problem is posed in the popular Independent Cascade model (IC) which, unlike the MCIC model, only considers a single campaign. The goal here is to compute a seed set S_{IM} of size k that maximizes the influence of S_{IM} in $\mathcal{G}$. In a small abuse of notation, this section refers to a possible world as the single deterministic graph $g \sim \mathcal{G}$ where each edge in $\mathcal{G}$ is associated with a single influence probability $p(e)$.

Borgs et al. [9] were the first to propose a novel method for solving the IM problem under the IC model that avoids the limitations of the original Monte Carlo-based solution [8]. Their approach, which was later refined by Tang et al. [10], is based on the concept of *Reverse Reachable (RR) sets* and is orders of magnitude faster than the greedy algorithm with Monte Carlo simulations, while still providing approximation guarantees with high probability. We follow the convention of [10] and refer to the method of [9] as *Reverse Influence Sampling (RIS)*. To explain how *RIS* works, Tang et al. [10] introduce the following definitions:

Definition 1 (Reverse Reachable Set). The reverse reachable set for a node v in $g \sim \mathcal{G}$ is the set of nodes that can reach v. (That is, for each node u in the RR set, there is a directed path from u to v in g.)

Definition 2 (Random RR Set). A random RR set is an RR set generated on an instance of $g \sim \mathcal{G}$, for a node selected uniformly at random from g.

Note, a random RR set encapsulates two levels of randomness: (i) a deterministic graph $g \sim \mathcal{G}$ is sampled where each edge $e \in E$ is independently removed with probability $(1 - p(e))$, and (ii) a "root" node v is randomly chosen from g. The connection between RR

sets and node activation is formalized in the following crucial lemma.

LEMMA 1. *[9]* For any seed set S and node v, the probability that an influence propagation process from S can activate v equals the probability that S overlaps an RR set for v.

Based on this result, the *RIS* algorithm runs in two steps:

1. Generate random RR sets from $\mathcal{G}$ until a threshold on the total number of steps taken has been reached.
2. Consider the maximum coverage problem of selecting k nodes to cover the maximum number of RR sets generated. Use the standard greedy algorithm for the problem to derive a $(1-1/e)$-approximate solution S_k^*. Return S_k^* as the seed set to use for activation.

The rationale behind *RIS* is as follows: if a node u appears in a large number of RR sets it should have a high probability to activate many nodes under the IC model; hence, u's expected influence should be large. As such, we can think of the number of RR sets u appears in as an estimator for u's expected influence. By the same reasoning, if a size-k node set S_k^* covers most RR sets, then S_k^* is likely to have the maximum expected influence among all size-k node sets in $\mathcal{G}$ leading to a good solution to the IM problem. As shown in [10], Lemma 1 is the key result that underpins the approximation guarantees of *RIS*.

The main contribution of Borgs et al. is an analysis of their proposed threshold-based approach: *RIS* generates RR sets until the total number of nodes and edges examined during the generation process reaches a pre-defined threshold Γ. Importantly, Γ must be set large enough to ensure a sufficient number of samples have been generated to provide a good estimator for expected influence. They show that when Γ is set to $\Theta((m+n)k\log n/\epsilon^2)$, *RIS* runs in near-optimal time $O((m+n)k\log n/\epsilon^2)$, and it returns a $(1-1/e-\epsilon)$-approximate solution to the IM problem with at least constant probability.

Due to the more complex dynamics involved in propagation under the MCIC model, adapting the reverse sampling approach to solve EIL is far from trivial.

4. NEW DEFINITIONS

In this section we introduce new definitions that are crucial to the development of our approach. In particular, we formalize the notion of *obstructed* nodes which is required to capture the necessary conditions for saving a node. Then, we use the notion of obstructed nodes to define RRC sets which are the analog to RR sets for the EIL problem.

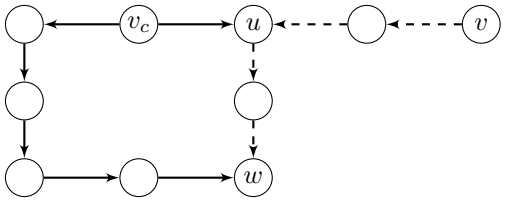

FIGURE 1: An example illustrating the concept of obstructed nodes where the possible world graph for campaign C is made up of the solid edges and the possible world for campaign L is made up of both solid and dashed lines.

Identifying Saved Nodes Given set A_L of vertices and (unweighted) directed graph $g \sim \mathcal{G}$, denote $SP_H(S,w)$ for a set S as the shortest path from any node $v \in S$ to w in graph H. For exposition purposes, we abuse notation and write $SP_H(\{v\},w)$ as $SP_H(v,w)$. Then, we denote $cl_g(A_L)$ as the set of nodes closer to A_L in G than to A_C in g. That is, a node $w \in cl_g(A_L)$ if there exists a node v such that $v \in A_L$ and $|SP_G(v,w)| \leq |SP_g(A_C,w)|$. When g is drawn from $\mathcal{G}$ this is a necessary, but not sufficient[3], condition for the set of nodes *saved* by A_L. We also require that the nodes in $cl_g(A_L)$ not be *obstructed* by the diffusion of campaign C in g.

DEFINITION 3 (Obstructed Nodes). A node $w \in cl_g(A_L)$ is *obstructed* and cannot be saved by A_L if for every path p from A_L to w there exists a node u on p such that $|SP_g(A_C,u)| < |SP_G(A_L,u)|$.

Let $obs_g(A_L)$ be the set of obstructed nodes for A_L. Conceptually, the nodes in $obs_g(A_L)$ are cutoff because some node on the paths from A_L is reached by campaign C before L which stops the diffusion of L.

To help illustrate the concept of obstructed nodes, consider the graph presented in Figure 1 and the following possible world instance. Assume that the solid lines are *live* edges that make up the deterministic graph $g \sim \mathcal{G}$ for campaign C in the influence propagation process. The dashed lines are edges that were not realized for campaign C. The adversary campaign C starts from v_c while the limiting campaign L starts from v. Recall, the deterministic graph G for campaign L in this possible world instance is comprised of *both* the solid and dashed edges due to the high effectiveness property. Observe that $|SP_G(v,w)| = 4$ and $|SP_g(A_C,w)| = 5$. However, w cannot be saved in the resulting cascade since at timestamp 1 the node u will adopt campaign C. This intersects the shortest path from v to w and therefore campaign L will not be able to reach node w since a node never switches campaigns. Thus, we say that node w is *obstructed* by C.

[3]In Budak et al.'s work, the set of nodes closer to A_L than A_C is established as a necessary and sufficient condition to *save* a node in the MCIC model, but we note that this should be revised to include our *obstructed* condition due to a gap in the proof of Claim 1 in [7].

TABLE 1: Frequently used notation.

Notation	Description
$\mathcal{G}$	a social network represented as a weighted directed graph $\mathcal{G}$
G, G_T	the underlying unweighted graph G and its transpose G_T constructed by reversing the direction of each edge
g	a possible world for campaign C obtained by sampling each edge $e \in \mathcal{G}$ independently with probability $p_C(e)$
n, m	the number of nodes and edges in $\mathcal{G}$ respectively
k	the size of the seed set for misinformation mitigation
C, L	the misinformation campaign C and the limiting campaign L
$p_C(e)$, $p_L(e)$	the propagation probability on an edge e for campaigns C and L respectively
$\pi(S)$	the prevention of a node set S in a misinformation propagation process on $\mathcal{G}$ (see Section 4)
$\omega(R)$, $\omega_\pi(R)$	the number of edges considered in generating an RRC set and that originate from nodes in an RRC set R (see Equation 4)
$\mathcal{R}$	the set of all RRC sets generated by Algorithm 1
$\mathcal{F}_\mathcal{R}(S)$	the fraction of RRC sets in $\mathcal{R}$ that are covered by a node set S
EPT	the expected width of a random RRC set
OPT_L	the maximum $\pi(S)$ for any size-k seed set S
KPT	Expected prevention of a seed set where seeds are chosen proportional to outdegree
λ	see Equation 5

Prevention & Saviours Next, we formally define the prevention, $\pi(A_L)$, which corresponds to the number of nodes saved by A_L. That is, $\pi(A_L) = |R_g(A_C) \cap (cl_g(A_L) \setminus obs_g(A_L))|$ where $R_H(S)$ is the set of nodes in graph H that are *reachable* from set S (a node v in H is reachable from S if there exists a directed path in H that starts from a node in S and ends at v). We write $\mathbb{E}[\pi(A_L)] = \mathbb{E}_{g\sim\mathcal{G}}[\pi(A_L)]$ for the expected prevention of A_L in $\mathcal{G}$. Finally, let $OPT_L = max_{S:|S|=k}\{\mathbb{E}[\pi(S)]\}$ be the maximum expected prevention of a set of k nodes.

We refer to the set of nodes that could have saved u as the *saviours* of u. A node w is a candidate saviour for u if there is a directed path from w to u in G (i.e. reverse reachability). Then, w is a saviour for u subject to the additional constraint that w would not be cutoff by the diffusion of A_C in g. That is, a candidate saviour w would be cutoff and cannot be a saviour for u if for every path p from w to u there exists a node v_b such that $|SP_g(A_C, v_b)| < |SP_G(w, v_b)|$. We refer to the set of candidate saviours for u that are cutoff as $\tau_g(u)$. Thus, we can define the saviours of u as the set $R_{G^T}(u)\setminus\tau_g(u)$. Therefore, we have:

Definition 4 (Reverse Reachability without Cutoff Set). The reverse reachability without cutoff (RRC) set for a node v in $g \sim \mathcal{G}$ is the set of saviour nodes of v, i.e. the set of nodes that can save v. (That is, for each node u in the RRC set, $u \in R_{G^T}(u) \setminus \tau_g(u)$.) If $v \notin R_g(A_C)$ then we define the corresponding RRC set as empty since v is not eligible to be saved.

Definition 5 (Random RRC Set). A random RRC set is an RRC set generated on an instance of $g \sim \mathcal{G}$, for a node selected uniformly at random from g.

Closing the Gap Before presenting our reverse sampling approach, we make the following remark regarding obstruction in the context of prior work. The key observation that lead to our definition of obstructed nodes is that the shortest path condition must hold along the *entire* path. This observation was missed by [7] in the MCIC model. Instead, a correct *recursive* definition was provided for the set of nodes that are saved, but the resulting characterization based on shortest paths misses the crucial case of nodes that are obstructed.

Importantly, the solution in [7] can be recovered with a modified proof for Claim 1 and Theorem 4.2. In particular, the statements must include the notion of obstructed nodes in their *inoculation graph* definition, but a careful inspection shows that their objective function remains submodular after this inclusion. As a result, the greedy approach of [7] still provides the stated approximation guarantees and also allows us to incorporate the ideas of [9] in our solution (as [9] requires a submodular objective function as well).

5. REVERSE PREVENTION SAMPLING

This section presents our misinformation prevention method *Reverse Prevention Sampling (RPS)* that employs the general reverse sampling framework. At a high level, *RPS*, in the same spirit as *RIS*, consists of two steps that parallel those described Section 3.2. The first step (Algorithm 3) leverages a key result that parallels Lemma 1 to establish a connection between RRC sets and the prevention process on $\mathcal{G}$ in order to derive a parameter θ that ensures a solution of high quality will be produced. In the second step, using the estimate θ from step one, it generates θ RRC sets (Algorithm 2) and then computes the maximum coverage on the resulting collection (Algorithm 1). More precisely, the two steps are:

1. **Parameter Estimation.** Compute a lower-

bound for the maximum expected prevention among all possible size-k seed sets for A_L and use the lower-bound to derive a parameter θ.

2. **Node Selection.** Sample θ random RRC sets from $\mathcal{G}$ to form a set $\mathcal{R}$ and then compute a size-k seed set S_k^* that covers a large number of RRC sets in $\mathcal{R}$. Return $A_L = S_k^*$ as the solution to the EIL problem.

In the rest of this section, we first tackle the challenging task of correctly generating RRC sets in the Node Selection step under the MCIC model. Next, we identify the conditions necessary for the Node Selection of *RPS* to return a solution of good quality and then describe how these conditions are achieved in the Parameter Estimation phase. Table 1 provides a reference to some of the frequently used notation.

Node Selection The pseudocode of *RPS*'s Node Selection step is presented in Algorithm 1. Given $\mathcal{G}$, k, A_C, and a constant θ as input, the algorithm stochastically generates θ random RRC sets, accomplished by repeated invocation of the prevention of misinformation process, and inserts them into a set $\mathcal{R}$. Next, the algorithm follows a greedy approach for the *maximum coverage problem* to select the final seed set A_L for the limiting campaign. In each iteration, the algorithm selects a node v_i that covers the largest number of RRC sets in $\mathcal{R}$, and then removes all those covered RRC sets from $\mathcal{R}$. The k selected nodes are put into a set S_k^*, which is then used to form the final seed set A_L for campaign L.

Algorithm 1 NodeSelection($\mathcal{G}$,k,A_C,θ)

1: $\mathcal{R} \leftarrow \emptyset$
2: Generate θ random RRC sets and insert them into $\mathcal{R}$.
3: Initialize a node set $S_k^* \leftarrow \emptyset$
4: **for** $i = 1,\ldots,k$ **do**
5: Identify the node v_i that covers the most RRC sets in $\mathcal{R}$
6: Add v_i into S_k^*
7: Remove from $\mathcal{R}$ all RRC sets that are covered by v_i
8: **return** S_k^*

Lines 4-8 in Algorithm 1 correspond to a standard greedy approach for a *maximum coverage problem*. The problem is equivalent to maximizing a submodular function with cardinality constraints for which it is well known that a greedy approach returns a $(1 - 1/e)$-approximate solution in linear time [68].

5.1. RRC set generation

Next, we describe how to generate RRC sets correctly for the EIL problem under the MCIC model, which is more complicated than generating RR sets for the IC model [10]. The construction of RRC sets is done according to Definition 4. Recall that in the MCIC model, whether a node can be saved or not is based on a number of factors such as whether v is reachable via a path in $g \sim \mathcal{G}$ from A_C and the diffusion history of each campaign. Our algorithms tackle the complex interactions between campaigns by first identifying nodes that can be influenced by C which reveals important information for generating RRC sets for L.

Line 2 generates $\mathcal{R}$ by repeated simulation of the misinformation prevention process. The generation of each random RRC set is implemented as two breath-first searches (BFS) on $\mathcal{G}$ and G^T respectively. The first BFS is a *forward labelling* process from A_C implemented as a forward BFS on $\mathcal{G}$ that computes the influence set of A_C in a possible world. The second BFS on G^T is a novel bounded-depth BFS with pruning that carefully tracks which nodes will become obstructed and is described in detail below.

Forward BFS with Lazy Sampling We first describe the forward labelling process. As the forward labeling is unlikely to reach the whole graph, we simply reveal edge states on demand ("lazy sampling"), based on the principle of deferred decisions. Given the seed set A_C of campaign C, we perform a randomized BFS starting from A_C where each outgoing edge e in $\mathcal{G}$ is traversed with $p_C(e)$ probability. The set of nodes traversed in this manner ($R_g(A_C)$) is equivalent to $I(A_C)$ for $g \sim \mathcal{G}$, due to deferred randomness. Note that in each step of the above BFS we record at each node w the minimum distance from A_C to w, denoted $D(w)$, for use in the second BFS.

Given a randomly selected node u in G, observe that for u to be able to be saved we require $u \in R_g(A_C)$. Therefore, if the randomly selected node $u \notin R_g(A_C)$ then we return an empty RRC set. On the other hand, if $u \in R_g(A_C)$, we have $D(u) = |SP_g(A_C, u)|$ as a result of the above randomized BFS which indicates the maximum distance from u that candidate saviour nodes can exist. We run a second BFS from u in G^T to depth $D(u)$ to determine the saviour nodes for u by carefully pruning those nodes that would become obstructed.

Bounded-depth BFS with Pruning The second BFS on G^T, presented in Algorithm 2, takes as input a source node u, the maximum depth $D(u)$, and a directed graph G^T. Algorithm 2 utilizes special indicator values associated with each node w to account for potential cutoffs from C. Each node w holds a variable, $\beta(w)$, which indicates the distance beyond w that the BFS can go before the diffusion would have been cutoff by C propagating in g. The β value for each node w is initialized to $D(w)$. In each round, the current node w has an opportunity to update the β value of each of its successors only if $\beta(w) > 0$. For each successor z of w, we assign $\beta(z) = \beta(w) - 1$ if $\beta(z) = \texttt{null}$ or if $\beta(z) > 0$

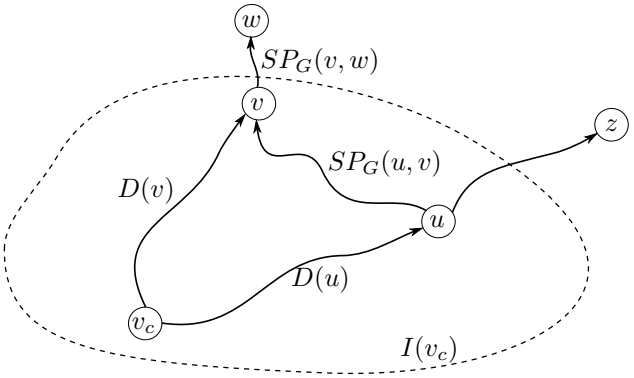

FIGURE 2: An overview of the primary scenarios encountered by Algorithm 2.

and $\beta(w)-1 < \beta(z)$. In this way, each ancestor of z will have an opportunity to apply a β value to z to ensure that if any ancestor has a β value then so will z and furthermore, the β variable for z will be updated with the smallest β value from its ancestors. We terminate the BFS early if we reach a node w with $\beta(w) = 0$.

Figure 2 captures the primary scenarios encountered by Algorithm 2 when initialized at u. The enclosing dotted line represents the extent of the influence of campaign C for the current influence propagation process. First, notice that if the BFS moves away from $A_C = \{v_c\}$, as in the case of node z, that, once we move beyond the influence boundary of C, there will be no potential for cutoff. As such, the BFS is free to traverse until the maximum depth $D(u)$ is reached. On the other hand, if the BFS moves towards (or perpendicular to) v_c then we must carefully account for potential cutoff. For example, when the BFS reaches v, we know the distance from v_c to v: $D(v) = SP_g(v_c, v)$. Therefore, the BFS must track the fact that there cannot exist saviours at a distance $D(v)$ beyond v. In other words, if we imagine initializing a misinformation prevention process from a node w such that $SP_G(v, w) > D(v)$ then v will adopt campaign C before campaign L can reach v. Therefore, at each out-neighbour of v we use the knowledge of $D(v)$ to track the distance beyond v that saviours can exist. This updating process tracks the smallest such value and is allowed to cross the enclosing influence boundary of campaign C ensuring that all potential for cutoff is tracked.

Finally, we collect all nodes visited during the process (including u), and use them to form an RRC set. The runtime of this procedure is precisely the sum of the degrees (in G) of the nodes in $R_g(A_C)$ plus the sum of the degrees of the nodes in $R_{G^T}(u) \setminus \tau(u)$.

We briefly note another key difference between *RPS* and *RIS* occurs in the RRC set generation step. Unlike in the single campaign setting, generating an RRC set is comprised of two phases instead of just one. First, we are required to simulate the spread of misinformation since being influenced by campaign C is a pre-condition for being saved. As a result, only a fraction of the simulation steps of *RPS* provide signal for the prevention value we are trying to estimate. This difference is made concrete in the running time analysis to follow.

Algorithm 2 generateRRC(u, $D(u)$, G^T)

```
1: let R ← ∅, Q be a queue and Q.enqueue(u)
2: set u.depth = 0 and label u as discovered
3: while Q is not empty do
4:     w ← Q.dequeue(), R ← R ∪ {w}
5:     if w.depth = D(u) OR β(w) = 0 then
6:         continue
7:     for all nodes z in G^T.adjacentEdges(w) do
8:         if β(w) > 0 AND β(z) > 0 then
9:             if β(w) − 1 < β(z) then
10:                β(z) ← β(w) − 1
11:        else if β(w) > 0 then
12:            β(z) ← β(w) − 1
13:        if z is not labelled as discovered then
14:            set z.depth = w.depth + 1, label z as
   discovered and Q.enqueue(z)
15: return R
```

5.2. Analysis

In this section we focus on two parameters: solution quality and runtime. For Algorithm 1 to return a solution with approximation guarantee, we will provide a lower bound on θ. Then, we will analyze the running time of the algorithm in terms of θ and a quantity EPT that captures the expected number of edges traversed when generating a random RRC set.

Approximation Guarantee We begin by establishing the crucial connection between RRC sets and the prevention process on $\mathcal{G}$. That is, the prevention of a set of nodes S is precisely n times the probability that a node u, chosen uniformly at random, has a saviour from S. Note, we say that a node set S covers or overlaps an RRC set R if $S \cap R \neq \emptyset$.

Lemma 2. For any seed set S and any node v, the probability that a prevention process from S can save v equals the probability that S overlaps an RRC set for v.

Proof. Let S be a fixed set of nodes, and v be a fixed node. Suppose that we generate an RRC set R for v on a graph $g \sim \mathcal{G}$. Let ρ_1 be the probability that S overlaps with R and let ρ_2 be the probability that S, when used as a seed set, can save v in a prevention process on $\mathcal{G}$. By Definition 4, if $v \in R_g(A_C)$ then ρ_1 equals the probability that a node $u \in S$ is a saviour for v. That is, ρ_1 equals the probability that G contains a directed path from $u \in S$ to v and $u \notin \tau(v)$ and 0 if $v \notin R_g(A_C)$. Meanwhile, if $v \in R_g(A_C)$ then ρ_2

equals the probability that a node $u \in S$ can save v (i.e. $v \in (cl_g(u) \setminus obs_g(u))$) and 0 if $v \notin R_g(A_C)$. It follows that $\rho_1 = \rho_2$ due to the equivalence between the set of saviours for v and the ability to save v. □

For any node set S, let $F_{\mathcal{R}}(S)$ be the fraction of RRC sets in $\mathcal{R}$ covered by S. Then, based on Lemma 2, we can prove that the expected value of $n \cdot F_{\mathcal{R}}(S)$ equals the expected prevention of S in $\mathcal{G}$.

COROLLARY 1. $\mathbb{E}[n \cdot F_{\mathcal{R}}(S)] = \mathbb{E}[\pi(S)]$

Proof. Observe that $\mathbb{E}[F_{\mathcal{R}}(S)]$ equals the probability that S intersects a random RRC set, while $\mathbb{E}[\pi(S)]/n$ equals the probability that a randomly selected node can be saved by S in a prevention process on $\mathcal{G}$. By Lemma 2, the two probabilities are equal, leading to $\mathbb{E}[n \cdot F_{\mathcal{R}}(S)] = \mathbb{E}[\pi(S)]$. □

Corollary 1 implies that we can estimate $\mathbb{E}[\pi(S)]$ by estimating the fraction of RRC sets in $\mathcal{R}$ covered by S. The number of sets covered by a node v in $\mathcal{R}$ is precisely the number of times we observed that v was a saviour for a randomly selected node u. We can therefore think of $n \cdot F_{\mathcal{R}}(S)$ as an estimator for $\mathbb{E}[\pi(S)]$. Our primary task is to show that it is a *good* estimator. Using Chernoff bounds, we show that $n \cdot F_{\mathcal{R}}(S)$ is an accurate estimator of any node set S's expected prevention, when θ is sufficiently large:

LEMMA 3. Suppose that θ satisfies

$$\theta \geq (8+2\epsilon)n \cdot \frac{l \log n + \log \binom{n}{k} + \log 2}{OPT_L \cdot \epsilon^2} \tag{1}$$

Then, for any set S of at most k nodes, the following inequality holds with at least $1 - n^{-l}/\binom{n}{k}$ probability:

$$\left| n \cdot F_{\mathcal{R}}(S) - \mathbb{E}[\pi(S)] \right| < \frac{\epsilon}{2} \cdot OPT_L \tag{2}$$

Proof. Let ρ be the probability that S overlaps with a random RRC set. Then, $\theta \cdot F_{\mathcal{R}}(S)$ can be regarded as the sum of θ i.i.d. Bernoulli variables with a mean ρ. By Corollary 1,

$$\rho = \mathbb{E}[F_{\mathcal{R}}(S)] = \mathbb{E}[\pi(S)]/n$$

Then, we have

$$\begin{aligned} &Pr\left[|n \cdot F_{\mathcal{R}}(S) - \mathbb{E}[\pi(S)]| \geq \frac{\epsilon}{2} \cdot OPT_L \right] \\ &= Pr\left[|\theta \cdot F_{\mathcal{R}}(S) - \rho\theta| \geq \frac{\epsilon\theta}{2n} \cdot OPT_L \right] \\ &= Pr\left[|\theta \cdot F_{\mathcal{R}}(S) - \rho\theta| \geq \frac{\epsilon \cdot OPT_L}{2n\rho} \cdot \rho\theta \right] \end{aligned} \tag{3}$$

Let $\delta = \epsilon \cdot OPT_L/(2n\rho)$. By the Chernoff bounds, Equation 1, and the fact that $\rho = \mathbb{E}[\pi(S)]/n \leq OPT_L/n$, we have

$$\begin{aligned} \text{r.h.s. of Eqn. 3} &< 2exp\Big(-\frac{\delta^2}{2+\delta} \cdot \rho\theta \Big) \\ &= 2exp\Big(-\frac{\epsilon^2 \cdot OPT_L^2}{8n^2\rho + 2\epsilon n \cdot OPT_L} \cdot \theta \Big) \\ &\leq 2exp\Big(-\frac{\epsilon^2 \cdot OPT_L^2}{8n \cdot OPT_L + 2\epsilon n \cdot OPT_L} \cdot \theta \Big) \\ &= 2exp\Big(-\frac{\epsilon^2 \cdot OPT_L}{(8+2\epsilon) \cdot n} \cdot \theta \Big) \leq \frac{1}{\binom{n}{k} \cdot n^l} \end{aligned}$$

Therefore, the lemma is proved. □

Based on Lemma 3, we prove that if Eqn. 1 holds, Algorithm 1 returns a $(1 - 1/e - \epsilon)$-approximate solution with high probability by a simple application of Chernoff bounds.

THEOREM 2. Given a θ that satisfies Equation 1, Algorithm 1 returns a $(1-1/e-\epsilon)$-approximate solution with at least $1 - n^{-l}$ probability.

Proof. Let S_k be the node set returned by Algorithm 1, and S_k^+ be the size-k node set that maximizes $F_{\mathcal{R}}(S_k^+)$ (i.e., S_k^+ covers the largest number of RRC sets in $\mathcal{R}$). As S_k is derived from $\mathcal{R}$ using a $(1-1/e)$-approximate algorithm for the maximum coverage problem, we have $F_{\mathcal{R}}(S_k) \geq (1-1/e) \cdot F_{\mathcal{R}}(S_k^+)$. Let S_k° be the optimal solution for the EIL problem on $\mathcal{G}$, i.e. $\mathbb{E}[\pi(S_k^\circ)] = OPT_L$. We have $F_{\mathcal{R}}(S_k^+) \geq F_{\mathcal{R}}(S_k^\circ)$, which leads to $F_{\mathcal{R}}(S_k) \geq (1-1/e) \cdot F_{\mathcal{R}}(S_k^\circ)$.

Assume that θ satisfies Equation 1. By Lemma 3, Equation 2 holds with at least $1 - n^{-l}/\binom{n}{k}$ probability for any given size-k node set S. Thus, by the union bound, Equation 2 should hold simultaneously for all size-k node sets with at least $1 - n^{-l}$ probability. In that case, we have

$$\begin{aligned} \mathbb{E}[\pi(S_k)] &> n \cdot F_{\mathcal{R}}(S_k) - \epsilon/2 \cdot OPT_L \\ &\geq (1-1/e) \cdot n \cdot F_{\mathcal{R}}(S_k^+) - \epsilon/2 \cdot OPT_L \\ &\geq (1-1/e) \cdot n \cdot F_{\mathcal{R}}(S_k^\circ) - \epsilon/2 \cdot OPT_L \\ &\geq (1-1/e) \cdot (1-\epsilon/2) \cdot OPT_L - \epsilon/2 \cdot OPT_L \\ &> (1-1/e-\epsilon) \cdot OPT_L \end{aligned}$$

Thus, the theorem is proved. □

Runtime First, we will define EPT which captures the expected number of edges traversed when generating a random RRC set. After that, we define the expected runtime of *RPS* in terms of EPT and the parameter θ.

Let M_R be the instance of $R_g(A_C)$ used in computing an RRC set R. Then, we define the *width* of an RRC set R, denoted as $\omega(R)$, as the number of edges in G that point to nodes in R plus the number of edges in G that originate from nodes in M_R. That is

$$\omega(R) = \sum_{u \in M_R} outdegree_G(u) + \sum_{v \in R} indegree_G(v) \tag{4}$$

Let EPT be the expected width of a random RRC set, where the expectation is taken over the randomness in R and M_R, and observe that Algorithm 1 has an expected runtime of $O(\theta \cdot EPT)$. This can be observed by noting that EPT captures the expected number of edge traversals required to generate a random RRC set since an edge is only considered in the propagation process (either of the two BFS's) if it points to a node in R or originates from a node in M_R. An important consideration is that, since OPT_L is unknown, we cannot set θ directly from Equation 1. For simplicity, we define

$$\lambda = (8+2\epsilon)n \cdot \Big(l \log n + \log \binom{n}{k} + \log 2\Big) \cdot \epsilon^{-2} \quad (5)$$

and rewrite Equation 1 as $\theta \geq \lambda / OPT_L$. In the parameter estimation step we employ techniques from [10], differing in several subtle ways due to the added complexity of the MCIC model, to derive a θ value for RPS that is above the threshold but also allows for practical efficiency.

5.3. Parameter Estimation

Our objective in this section is to identify a θ that makes $\theta \cdot EPT$ reasonably small, while still ensuring $\theta \geq \lambda / OPT_L$. We begin with some definitions. Let $\mathcal{V}^*$ be a probability distribution over the nodes in G, such that the probability mass for each node is proportional to its indegree in G. Let v^* be a random variable following $\mathcal{V}^*$ and recall that M_R is a random instance of $R_g(A_C)$ that is equivalent to the influence $I(A_C)$ for a possible world g. We define the *prewidth of* R, denoted $\omega_C(R)$, as the number of edges in G that originate from nodes in M_R, i.e. $\omega_C(R) = \sum_{u \in M_R} outdegree_G(u)$. Then we prove the following.

Lemma 4. $\frac{m}{n} \cdot \mathbb{E}[\pi(\{v^*\})] = EPT - \mathbb{E}[\omega_C(R)]$, where the expectation of $\pi(\{v^*\})$ and $\omega_C(R)$ is taken over the randomness in v^* and the prevention process.

Proof. Let R be a random RRC set, M_R be the random instance of $R_g(A_C)$ used to compute R, and p_R be the probability that a randomly selected edge from G points to a node in R. Then, $EPT = \mathbb{E}[\omega_C(R)] + \mathbb{E}[p_R \cdot m]$, where the expectation is taken over the random choices of R and M_R.

Let v^* be a sample from $\mathcal{V}^*$ and $b(v^*, R)$ be a boolean function that returns 1 if $v^* \in R$, and 0 otherwise. Then, for any fixed R, $p_R = \sum_{v^*}(\Pr[v^*] \cdot b(v^*, R))$. Now consider that we fix v^* and vary R. Define $p_{v^*,R} = \sum_R (\Pr[R] \cdot b(v^*, R))$ so that by Lemma 2, $p_{v^*,R}$ equals the probability that a randomly selected node can be saved in a prevention process when $\{v^*\}$ is used as a seed set. Therefore, $\mathbb{E}[p_{v^*,R}] = \mathbb{E}[\pi(\{v^*\})]/n$. This leads to

$$\begin{aligned}
\mathbb{E}[p_R] &= \sum_R (\Pr[R] \cdot p_R) \\
&= \sum_R (\Pr[R] \cdot \sum_{v^*} (\Pr[v^*] \cdot b(v^*, R))) \\
&= \sum_{v^*} (\Pr[v^*] \cdot \sum_R (\Pr[R] \cdot b(v^*, R))) \\
&= \sum_{v^*} (\Pr[v^*] \cdot p_{v^*,R}) \\
&= \mathbb{E}[p_{v^*,R}] = \mathbb{E}[\pi(\{v^*\})]/n
\end{aligned}$$

Therefore, $EPT = \mathbb{E}[\omega_C(R)] + m \cdot \mathbb{E}[p_R] = \mathbb{E}[\omega_C(R)] + \frac{m}{n} \cdot \mathbb{E}[\pi(\{v^*\})]$. This completes the proof. □

Lemma 4 shows that if we randomly sample a node from $\mathcal{V}^*$ and calculate its expected prevention p, then on average we have $p = \frac{n}{m}(EPT - \mathbb{E}[\omega_C(R)])$. This implies that $\frac{n}{m}(EPT - \mathbb{E}[\omega_C(R)]) \leq OPT_L$, since OPT_L is the maximum expected prevention of any size-k node set. Importantly, the expected prevention of a randomly sampled node from $\mathcal{V}^*$ is not solely defined by the expected number of edges traversed when generating an RRC set. Instead, we must accurately account for those edges that were traversed in simulating the spread of campaign C. Unlike the work of [10], where every edge traversed in generating an RR set provides signal for the quantity they are trying to estimate (influence), our setting must balance the fact that only a subset of edges traversed will provide signal for the quantity we are trying to estimate, namely prevention. This distinction manifests itself in the final runtime expression that follows.

Recall that the expected runtime complexity of Algorithm 1 is $O(\theta \cdot EPT)$. Now, suppose we are able to identify a parameter t such that $t = \Omega(\frac{n}{m}(EPT - \mathbb{E}[\omega_C(R)]))$ and $t \leq OPT_L$. Then, by setting $\theta = \lambda / t$, we can guarantee that Algorithm 1 is correct, since $\theta \geq \lambda / OPT_L$, and has an expected runtime complexity of

$$O(\theta \cdot EPT) = O\Big(\frac{\lambda}{t} \cdot EPT\Big) = O\left(\frac{\lambda \cdot EPT}{\frac{n}{m}(EPT - \mathbb{E}[\omega_C(R)])}\right) \quad (6)$$

Furthermore, if we define a ratio $\gamma \in (0, 1)$ which captures the relationship between $\mathbb{E}[\omega_C(R)]$ and EPT by writing $\mathbb{E}[\omega_C(R)] = \gamma EPT$, we can rewrite Equation 6 as

$$O\Big(\frac{m}{n}\Big(\frac{1}{1-\gamma}\Big)\lambda\Big) = O((k+l)(m+n)(1/(1-\gamma)) \log n / \epsilon^2) \quad (7)$$

Note that γ is a data-dependent approximation factor not present in [10], but arises from the MCIC model. In particular, the RRC set generation relies crucially on first computing the spread of misinformation from

campaign C in order to determine the set of nodes that can be saved. See Section 6 for a detailed discussion of γ.

Computing t Ideally, we seek a t that increases monotonically with k to mimic the behaviour of OPT_L. Suppose we take k samples $e_i = (v_i, w_i)$ with replacement over a uniform distribution on the edges in G, and use the v_i's to form a node set S^*. Let KPT be the mean of the expected prevention of S^* over the randomness in S^* and the prevention process. Due to the submodularity of the prevention function, it can be verified that KPT increases with k and

$$\frac{n}{m}\Big(EPT - \mathbb{E}[\omega_C(R)]\Big) \leq KPT \leq OPT_L \qquad (8)$$

Additionally,

LEMMA 5. Let R be a random RRC set and define the *subwidth* of R ($\omega_\pi(R)$), the number of edges in G that point to nodes in R, as $\omega_\pi(R) = \sum_{v \in R} indegree_G(v)$. Define

$$\kappa(R) = 1 - \left(1 - \frac{\omega_\pi(R)}{m}\right)^k \qquad (9)$$

Then, $KPT = n \cdot \mathbb{E}[\kappa(R)]$, where the expectation is taken over the random choices of R.

Proof. Let S^* be a node set formed by the v_i from k samples $e_i = (v_i, w_i)$ over a uniform distribution on the edges in G, with duplicates removed. Let R be a random RRC set, and α_R be the probability that S^* overlaps with R. Then, by Corollary 1,

$$KPT = \mathbb{E}[\pi(S^*)] = \mathbb{E}[n \cdot \alpha_R]$$

Consider that we sample k times over a uniform distribution on the edges in G. Let E^* be the set of edges sampled, with duplicates removed. Let α'_R be the probability that one of the edges in E^* points to a node in R. It can be verified that $\alpha'_R = \alpha_R$. Furthermore, given that there are $\omega_\pi(R)$ edges in G that point to nodes in R, $\alpha'_R = 1 - (1 - \omega_\pi(R)/m)^k = \kappa(R)$. Therefore,

$$KPT = \mathbb{E}[n \cdot \alpha_R] = \mathbb{E}[n \cdot \alpha'_R] = \mathbb{E}[n \cdot \kappa(R)]$$

Which proves the lemma. □

Lemma 5 shows we can estimate KPT by computing $n \cdot \kappa(R)$ on a set of random RRC sets and averaging over a sufficiently large sample size. However, if we want to obtain an estimate of KPT with $\delta \in (0, 1)$ relative error with at least $1 - n^{-l}$ probability then Chernoff bounds dictate that the number of samples required is $\Omega(ln \log n \cdot \delta^{-2}/KPT)$ which depends on KPT itself. This issue is also encountered in [10] and we are able to resolve it by mimicking their adaptive sampling approach, which dynamically adjusts the number of measurements based on the observed values of $\kappa(R)$, under the MCIC model.

Algorithm 3 KptEstimation($\mathcal{G}$,k,A_C)

1: **for** $i = 1$ to $2\log_2 n - 1$ **do**
2: Let $c_i = (6l \log n + 6\log(\log_2 n)) \cdot 2^i$
3: Let $sum = 0$
4: **for** $j = 1$ to c_i **do**
5: Generate a random RRC set R
6: $\kappa(R) = 1 - (1 - \frac{\omega_\pi(R)}{m})^k$
7: $sum = sum + \kappa(R)$
8: **if** $sum/c_i > 1/2^i$ **then**
9: **return** $KPT^* = n \cdot sum/(2 \cdot c_i)$
10: **return** $KPT^* = 1/n$

Estimating KPT The sampling approach for estimating KPT is presented in Algorithm 3. We begin with a high level description of how the algorithm proceeds. Since KPT is an unknown quantity, we begin with the assumption that it takes on the value $n/2$. Then, we compute an estimate for the expected value of $\kappa(R)$ based on a relatively few number of samples. Chernoff bounds allow us to determine if the computed value of $KPT = n \cdot \kappa(R)$ is a good estimator and, if so, the algorithm terminates. However, if the estimate is much smaller than $n/2$ we apply the standard doubling approach and generate an increased number of samples to determine if KPT takes on a value close to half the initial estimate. The algorithm continues computing estimates for KPT, based on an increasing number of samples, and comparing to values that halve in size until the error bounds dictated by Chernoff bounds indicate we have reached a suitably precise estimation of KPT.

In each iteration (Lines 2-7), the goal of Algorithm 3 is to compute the average value of $\kappa(R)$ over c_i randomly generated RRC sets from $\mathcal{G}$. As described in Lemma's 6 and 7 below, the c_i are chosen carefully such that if the average computed for $\kappa(R)$ over the c_i samples is greater than 2^{-i} then we can conclude that we have a good estimate for KPT with high probability. More precisely, that the expected value of $\kappa(R)$ is at least half of the average computed. Conversely, if the average computed is too small then Chernoff bounds imply that we have a bad estimate for KPT and the algorithm proceeds to the next iteration.

The IM problem benefits from a lower bound on KPT of 1 which allows the algorithm to terminate in $\log_2 n - 1$ iterations. However, we do not benefit from this lower bound for the MCIC problem, since a seed node for campaign C is not guaranteed to save any nodes in M_R. In the case that the true value of KPT is very small, the algorithm will terminate in the $2\log_2 n$-th iteration and return $KPT^* = 1/n$, which corresponds to the smallest possible KPT under the mild assumption that $|M_R| \geq 2$ always holds. This lower bound follows directly from the probability that

a randomly selected seed used for computing KPT falls within M_R. As we will show in the next section, $KPT^* \in [KPT/4, OPT_L]$ holds with a high probability and therefore setting $\theta = \lambda/KPT^*$ ensures Algorithm 1 is correct and achieves the expected runtime complexity in Equation 7.

Performance Bounds Proving the correctness and demonstrating bounds on the runtime for Algorithm 3 requires a careful analysis of the algorithm's behaviour. As shown below, we make use of two lemmas to prove that the algorithm's estimate of KPT^* is close to KPT.

Let $\mathcal{K}$ be the distribution of $\kappa(R)$ over random RRC sets in $\mathcal{G}$ with domain $[0, 1]$. Let $\mu = KPT/n$, and s_i be the sum of c_i i.i.d. samples from $\mathcal{K}$, where c_i is defined as $c_i = (6l \log n + 6 \log(\log_2 n)) \cdot 2^i$. Chernoff bounds give

Lemma 6. If $\mu \leq 2^{-j}$, then for any $i \in [1, j-1]$,

$$Pr\left[\frac{s_i}{c_i} > \frac{1}{2^i}\right] < \frac{1}{n^l \cdot \log_2 n} \tag{10}$$

Proof. Let $\delta = (2^{-i} - \mu)/\mu$. By the Chernoff bounds,

$$\begin{aligned} Pr\left[\frac{s_i}{c_i} > 2^{-i}\right] &\leq exp\left(-\frac{\delta^2}{2+\delta} \cdot c_i \cdot \mu\right) \\ &= exp(-c_i \cdot (2^{-i} - \mu)^2/(2^{-i} + \mu)) \\ &\leq exp(-c_i \cdot 2^{-i-1}/3) = \frac{1}{n^l \cdot \log_2 n} \end{aligned}$$

This completes the proof. □

By Lemma 6, if $KPT \leq 2^{-j}$, then Algorithm 3 is very unlikely to terminate in any of the first $j-1$ iterations. This prevents the algorithm from outputting a KPT^* too much larger than KPT.

Lemma 7. If $\mu \geq 2^{-j}$, then for any $i \geq j+1$,

$$Pr\left[\frac{s_i}{c_i} > \frac{1}{2^i}\right] > 1 - \left(\frac{1}{n^l \cdot \log_2 n}\right)^{2^{i-j-1}} \tag{11}$$

Proof. Let $\delta = (\mu - 2^{-i})/\mu$. By the Chernoff bounds,

$$\begin{aligned} Pr\left[\frac{s_i}{c_i} \leq 2^{-i}\right] &\leq exp\left(-\frac{\delta^2}{2} \cdot c_i \cdot \mu\right) \\ &= exp(-c_i \cdot (\mu - 2^{-i})^2/(2 \cdot \mu)) \\ &\leq exp(-c_i \cdot \mu/8) < \left(\frac{1}{n^l \cdot \log_2 n}\right)^{2^{i-j-1}} \end{aligned}$$

This completes the proof. □

By Lemma 7, if $KPT \leq 2^{-j}$ and Algorithm 3 enters iteration $i > j+1$, then it will terminate in the i-th iteration with high probability. This ensures that the algorithm does not output a KPT^* that is too much smaller than KPT.

Based on Lemmas 6 and 7, we prove the following theorem for the correctness and expected runtime of Algorithm 3.

Theorem 3. When $n \geq 2$ and $l \geq 1/2$, Algorithm 3 returns $KPT^* \in [KPT/4, OPT_L]$ with at least $1 - 2n^{-l}$ probability, and runs in $O(l(m+n)(1/(1-\gamma)) \log n)$ expected time. Furthermore, $\mathbb{E}[\frac{1}{KPT^*}] < \frac{12}{KPT}$.

Proof. Assume that $KPT/n \in [2^{-j}, 2^{-j+1}]$. We first prove the accuracy of the KPT^* returned by Algorithm 3.

By Lemma 6 and the union bound, Algorithm 3 terminates no later than the $(j-2)$-th iteration with less than $n^{-l}(j-2)/\log_2 n$ probability. Conversely, if Algorithm 3 reaches the $(j+1)$-th iteration, then by Lemma 7, it terminates in the $(j+1)$-th iteration with at least $1 - n^{-l}/\log_2 n$ probability. Applying the union bound and noting that Algorithm 3 has at most $2\log_2 n - 1$ iterations, Algorithm 3 should terminate in the $(j-1)$-th, j-th, or $(j+1)$-th iteration with probability at least $1 - n^{-l}(2\log_2 n - 2)/\log_2 n$. As a result, KPT^* must be larger than $n/2 \cdot 2^{-j-1}$, giving $KPT^* > KPT/4$. Furthermore, KPT^* should be $n/2$ times the average of at least c_{j-1} i.i.d. samples from $\mathcal{K}$. Then, Chernoff bounds yield

$$Pr[KPT^* \geq KPT] \leq n^{-l}/\log_2 n$$

Applying the union bound again, Algorithm 3 returns, with probability at least $1 - 2n^{-l}$ probability, $KPT^* \in [KPT/4, KPT] \subseteq [KPT/4, OPT_L]$.

Next, we analyze the expected runtime of Algorithm 3. Recall that the i-th iteration of the algorithm generates c_i RRC sets, and each RRC set takes $O(EPT)$ expected time. Given that $c_{i+1} = 2 \cdot c_i$ for any i, the first $j+1$ iterations generate less than $2 \cdot c_{j+1}$ RRC sets in total. Meanwhile, for any $i' \geq j+2$, Lemma 7 shows that Algorithm 3 has at most $n^{-l \cdot 2^{i'-j-1}}/\log_2 n$ probability to reach the i'-th iteration. Therefore, when $n \geq 2$ and $l \geq 1/2$, the expected number of RRC sets generated after the first $j+1$ iterations is less than

$$\sum_{i'=j+2}^{2\log_2 n - 1} \left(c_{i'} \cdot \left(\frac{1}{n^l \cdot \log_2 n}\right)^{2^{i-j-1}}\right) < c_{j+2}$$

Hence, the expected total number of RRC sets generated by Algorithm 3 is less than $2c_{j+1} + c_{j+2} = 2c_{j+2}$. Therefore, the expected time complexity of the algorithm is

$$
\begin{aligned}
O(c_{j+2}\cdot EPT) &= O(2^j l \log n \cdot EPT) \\
&= O\Big(2^j l \log n \cdot \Big(1+\frac{m}{n}\Big)\cdot KPT \cdot \Big(\frac{1}{1-\gamma}\Big)\Big) \\
&= O\Big(2^j l \log n \cdot (m+n)\cdot 2^{-j} \cdot \Big(\frac{1}{1-\gamma}\Big)\Big) \\
&= O\Big(l \log n \cdot (m+n)\cdot \Big(\frac{1}{1-\gamma}\Big)\Big)
\end{aligned}
$$

Here we used Equation 8 in the second equality. Finally, we show that $\mathbb{E}[1/KPT^*] < 12/KPT$. Observe that if Algorithm 3 terminates in the i-th iteration, it returns $KPT^* \geq n \cdot 2^{-i-1}$. Let ζ_i denote the event that Algorithm 3 stops in the i-th iteration. By Lemma 7, when $n \geq 2$ and $l \geq 1/2$, we have

$$
\begin{aligned}
\mathbb{E}[1/KPT^*] &= \sum_{i=1}^{2\log_2 n-1} \Big(2^{i+1}/n \cdot \Pr[\zeta_i]\Big) \\
&< \sum_{i=j+2}^{2\log_2 n-1} \Big(2^{i+1}/n \cdot \Big(n^{-l\cdot 2^{i-j-1}}/\log_2 n\Big)\Big) + 2^{j+2}/n \\
&< (2^{j+3}+2^{j+2})/n \leq 12/KPT
\end{aligned}
$$

This completes the proof. □

5.4. Improved Parameter Estimation

This section describes a new heuristic for improving the practical performance of *RPS*, without affecting its asymptotic guarantees, by improving the estimated lower bound on OPT_L. Our heuristic simplifies the one introduced in [10] and further, is adapted to the MCIC setting.

Algorithm 4 RefineKPT($\mathcal{G}$,k,A_C, KPT^*, ϵ')

1: Let $\lambda' = (2+\epsilon')ln\log n \cdot (\epsilon')^{-2}$.
2: Let $\theta' = \lambda'/KPT^*$.
3: Generate θ' random RRC sets; put them into a set $\mathcal{R}'$.
4: Initialize a node set $S'_k = \emptyset$.
5: **for** $i = 1$ to k **do**
6: Identify node v_i that covers the most RRC sets in $\mathcal{R}'$.
7: Add v_i to S'_k.
8: Remove from $\mathcal{R}'$ all RRC sets that are covered by v_i.
9: Let f be the fraction of the original θ' RRC sets that are covered by S'_k.
10: Let $KPT' = f \cdot n/(1+\epsilon')$
11: **return** $KPT^+ = max\{KPT', KPT^*\}$

Observe that the KPT^* output by Algorithm 3 largely determines the efficiency of *RPS*. If KPT^* is close to OPT_L, then $\theta = \lambda/KPT^*$ is small and Algorithm 1 only needs to generate a relatively small number of RRC sets. However, if $KPT^* \ll OPT_L$ then the efficiency of Algorithm 1 degrades rapidly and, in turn, so does the overall performance of *RPS*.

To remedy this issue, we can add an intermediate step before Algorithm 1 to refine KPT^* into a potentially tighter lower-bound of OPT_L. The intuition behind this heuristic is to generate a reduced number θ' of random RRC sets, placing them into a set $\mathcal{R}'$, and then apply the greedy approach (for the maximum coverage problem) on $\mathcal{R}'$ to obtain a good estimator for the maximum expected prevention in $\mathcal{R}'$. Thus, we can use the estimation as a good lower-bound for OPT_L.

Algorithm 4 shows the pseudo-code of the intermediate step. It first generates θ' random RRC sets and invokes the greedy approach for the maximum coverage problem on $\mathcal{R}'$ to obtain a size-k node set S'_k. Algorithm 4 computes the fraction f of RRC sets that are covered by S'_k so that, by Corollary 1, $f \cdot n$ is an unbiased estimation of $\mathbb{E}[\pi(S'_k)]$. We set θ' based on the KPT^* output by Algorithm 3 to a reasonably large number to ensure that $f \cdot n < (1+\epsilon') \cdot \mathbb{E}[\pi(S'_k)]$ occurs with at least $1-n^{-l}$ probability. Based on this, Algorithm 4 computes $KPT' = f \cdot n/(1+\epsilon')$ scaling $f \cdot n$ down by a factor of $1+\epsilon'$ to ensure that $KPT' \leq \mathbb{E}[\pi(S'_k)] \leq OPT_L$. The final output of Algorithm 4 is $KPT^+ = max\{KPT', KPT^*\}$. Below we give a lemma that shows the theoretical guarantees of Algorithm 4.

LEMMA 8. Given that $\mathbb{E}[\frac{1}{KPT^*}] < \frac{12}{KPT}$, Algorithm 4 runs in $O(l(m+n)(1/(1-\gamma))\log n/(\epsilon')^2)$ expected time. In addition, it returns $KPT^+ \in [KPT^*, OPT_L]$ with at least $1-n^{-l}$ probability, if $KPT^* \in [KPT/4, OPT_L]$.

Proof. We first analyze the expected time complexity of Algorithm 4. Observe that the expected time complexity of Lines 1-3 of Algorithm 4 is $O(\mathbb{E}[\frac{\lambda'}{KPT^*}] \cdot EPT)$, since they generate $\frac{\lambda'}{KPT^*}$ random RRC sets, each of which takes $O(EPT)$ expected time to generate. By Theorem 3, $\mathbb{E}[\frac{1}{KPT^*}] < \frac{12}{KPT}$. Therefore,

$$
\begin{aligned}
O\Big(\mathbb{E}\Big[\frac{\lambda'}{KPT^*}\Big]\cdot EPT\Big) &= O\Big(\frac{\lambda'}{KPT}\cdot EPT\Big) \\
&= O\Big(\frac{\lambda'}{KPT}\cdot\Big(1+\frac{m}{n}\Big)\cdot KPT\cdot\Big(\frac{1}{1-\gamma}\Big)\Big) \\
&= O\Big(\lambda'\cdot\Big(1+\frac{m}{n}\Big)\cdot\Big(\frac{1}{1-\gamma}\Big)\Big) \\
&= O(l(m+n)(1/(1-\gamma))\log n/(\epsilon')^2)
\end{aligned}
$$

On the other hand, Lines 4-12 run in time linear to the total size of the RRC sets in $\mathcal{R}'$, i.e. the set of all RRC sets generated in Lines 1-3 of Algorithm 4. Given that the expected total size of the RRC sets in $\mathcal{R}'$ should be no more than $O(l(m+n)(1/(1-\gamma))\log n)$, Lines 4-12 of Algorithm 4 have an expected time complexity of $O(l(m+n)(1/(1-\gamma))\log n)$. Therefore, the expected

time complexity of Algorithm 4 is $O(l(m+n)(1/(1-\gamma))\log n/(\epsilon')^2)$.

Next, we prove that Algorithm 4 returns $KPT^+ \in [KPT^*, OPT_L]$ with high probability. First, observe that $KPT^+ \geq KPT^*$ trivially holds, as Algorithm 4 sets $KPT^+ = max\{KPT', KPT^*\}$, where KPT' is derived in Line 11 of Algorithm 4. To show that $KPT^+ \in [KPT^*, OPT_L]$, it suffices to prove that $KPT' \leq OPT_L$.

By Line 11 of Algorithm 4, $KPT' = f \cdot n/(1+\epsilon')$, where f is the fraction of RRC sets in $\mathcal{R}'$ that is covered by S'_k, where $\mathcal{R}'$ is a set of θ' random RRC sets, and S'_k is a size-k node set generated from Lines 4-9 in Algorithm 4. Therefore, $KPT' \leq OPT_L$ if and only if $f \cdot n \leq (1+\epsilon') \cdot OPT_L$.

Let ρ' be the probability that a random RRC set is covered by S'_k. By Corollary 1, $\rho' = \mathbb{E}[\pi(S'_k)]/n$. In addition, $f \cdot \theta'$ can be regarded as the sum of θ' i.i.d. Bernoulii variables with a mean ρ'. Therefore, we have

$$\begin{aligned}
&\Pr[f \cdot n > (1+\epsilon') \cdot OPT_L] \\
&\quad \leq \Pr\left[n \cdot f - \mathbb{E}[\pi(S'_k)] > \epsilon' \cdot OPT_L\right] \\
&\quad = \Pr\left[\theta' \cdot f - \theta' \cdot \rho' > \frac{\theta'}{n} \cdot \epsilon' \cdot OPT_L\right] \\
&\quad = \Pr\left[\theta' \cdot f - \theta' \cdot \rho' > \frac{\epsilon' \cdot OPT_L}{n \cdot \rho'} \cdot \theta' \cdot \rho'\right]
\end{aligned} \tag{12}$$

Let $\delta = \epsilon' \cdot OPT_L/(n\rho')$. By the Chernoff bounds, we have

$$\begin{aligned}
\text{r.h.s. of Eqn. 12} &\leq \exp\left(-\frac{\delta^2}{2+\delta} \cdot \rho'\theta'\right) \\
&= \exp\left(-\frac{\epsilon'^2 \cdot OPT_L^2}{2n^2\rho' + \epsilon' n \cdot OPT_L} \cdot \theta'\right) \\
&\leq \exp\left(-\frac{\epsilon'^2 \cdot OPT_L^2}{2n \cdot OPT_L + \epsilon' n \cdot OPT_L} \cdot \theta'\right) \\
&= \exp\left(-\frac{\epsilon'^2 \cdot OPT_L}{(2+\epsilon') \cdot n} \cdot \frac{\lambda'}{KPT^*}\right) \\
&\leq \exp\left(-\frac{\epsilon'^2 \cdot \lambda'}{(2+\epsilon') \cdot n}\right) \leq \frac{1}{n^l}
\end{aligned}$$

Therefore, $KPT' = f \cdot n/(1+\epsilon') \leq OPT_L$ holds with at least $1 - n^{-l}$ probability. This completes the proof. □

Note that the time complexity of Algorithm 4 is smaller than that of Algorithm 3 by a factor of k, since the former only needs to accurately estimate the expected prevention of one node set (i.e. S'_k), whereas the latter needs to ensure accurate estimations for $\binom{n}{k}$ node sets simultaneously. Additionally, our new approach eliminates the need to first compute a seed set from the RRC sets generated in the last iteration of Algorithm 3 as in [10].

Wrapping Up In summary, we integrate Algorithm 4 into *RPS* and obtain an improved solution (referred to as RPS^+) as follows. Given $\mathcal{G}$, k, A_C, ϵ, and l, we first invoke Algorithm 3 to derive KPT^*. After that, we feed $\mathcal{G}$, k, A_C, KPT^*, and a parameter ϵ^* (as defined in [10]) to Algorithm 4, and obtain KPT^+ in return. Then, we compute $\theta = \lambda/KPT^+$, where λ is as defined in Equation 5. Finally, we run Algorithm 1 with $\mathcal{G}$, k, A_C, and θ as the input and get the output S^*_k as the final result of prevention maximization.

By Theorems 2 and 3, Equation 7, and the union bound, *RPS* runs in $O((k+l)(m+n)(1/(1-\gamma))\log n/\epsilon^2)$ expected time and it can be verified that when $\epsilon' \geq \epsilon/\sqrt{k}$, RPS^+ has the same time complexity as *RPS*. Furthermore, RPS^+ returns a $(1-1/e-\epsilon)$-approximate solution with at least $1 - 4n^{-l}$ probability and the success probability can be increased to $1-n^{-l}$ by scaling l up by a factor of $1 + \log 4/\log n$.

Finally, we note that the time complexity of *RPS* is *near-optimal* up to the instance-specific factor γ under the MCIC model, as it is only a $(\frac{1}{1-\gamma})\log n$ factor larger than the $\Omega(m+n)$ lower-bound proved in Section 6 (for fixed k, l, and ϵ).

6. LOWER BOUNDS

Comparison with MCGreedy *MCGreedy* runs in $O(kmnr)$ time, where r is the number of Monte Carlo samples used to estimate the expected prevention of each node set. Budak et al. do not provide a detailed analysis related to how r should be set to achieve a $(1-1/e-\epsilon)$-approximation ratio in the MCIC model, only pointing out that when each estimation of expected prevention has ϵ relative error, *MCGreedy* returns a $(1-1/e-\epsilon')$-approximate solution for a particular ϵ' [7]. In the following lemma, we present a more precise characterization of the relationship between r and *MCGreedy*'s approximation ratio in the MCIC model.

Lemma 9. *MCGreedy* returns a $(1-1/e-\epsilon)$-approximate solution with at least $1-n^{-l}$ probability, if

$$r \geq (8k^2 + 2k\epsilon) \cdot n \cdot \frac{(l+1)\log n + \log k}{\epsilon^2 \cdot OPT_L} \tag{13}$$

Proof. Let S be any node set that contains no more than k nodes in G, and $\xi(S)$ be an estimation of $\mathbb{E}[\pi(S)]$ using r Monte Carlo steps. We first prove that, if r satisfies Equation 13, then $\xi(S)$ will be close to $\mathbb{E}[\pi(S)]$ with a high probability.

Let $\mu = \mathbb{E}[\pi(S)]/n$ and $\delta = \epsilon OPT_L/(2kn\mu)$. By the Chernoff bounds, we have

$$\begin{aligned}
&\Pr\left[|\xi(S) - \mathbb{E}[\pi(S)]| > \frac{\epsilon}{2k} OPT_L\right] \\
&= \Pr\left[\left|r \cdot \frac{\xi(S)}{n} - r \cdot \frac{\mathbb{E}[\pi(S)]}{n}\right| > \frac{\epsilon}{2kn} \cdot r \cdot OPT_L\right] \\
&= \Pr\left[\left|r \cdot \frac{\xi(S)}{n} - r \cdot \frac{\mathbb{E}[\pi(S)]}{n}\right| > \delta \cdot r \cdot \mu\right] \\
&< 2\exp\left(-\frac{\delta^2}{2+\delta} \cdot r \cdot \mu\right) \\
&= 2\exp\left(-\frac{\epsilon^2}{(8k^2 + 2k\epsilon) \cdot n} \cdot r \cdot \mu\right) \\
&= 2\exp((l+1)\log n + \log k) \\
&= \frac{1}{k \cdot n^{l+1}} \qquad (14)
\end{aligned}$$

Observe that, given $\mathcal{G}$, k, and A_C *MCGreedy* runs in k iterations, each of which estimates the expected prevention f at most n node sets with sizes no more than k. Therefore, the total number of node sets inspected by *MCGreedy* is at most kn. By Equation 14 and the union bound, with at least $1 - n^{-l}$ probability, we have

$$|\xi(S') - \mathbb{E}[\pi(S')]| \leq \frac{\epsilon}{2k} OPT_L \qquad (15)$$

for all those kn node sets S' simultaneously. In what follows, we analyze the accuracy of *MCGreedy*'s output, under the assumption that for any node set S' considered by *MCGreedy*, it obtains a sample of $\xi(S')$ that satisfies Equation 14. For convenience, we abuse notation and use $\xi(S')$ to denote the aforementioned sample.

Let $S_0 = \emptyset$, and S_i ($i \in [1, k]$) be the node set selected by *MCGreedy* in the i-th iteration. We define $x_i = OPT_L - \pi(S_i)$, and $y_i(v) = \pi(S_{i-1} \cup \{v\}) - \pi(s_{i-1}))$ for any node v. Let v_i be the node that maximizes $y_i(v_i)$. Then, $y_i(v_i) \geq x_{i-1}/k$ must hold; otherwise, for any size-k node set S, we have

$$\begin{aligned}
\pi(S) &\leq \pi(S_{i-1}) + \pi(S \setminus S_{i-1}) \\
&\leq \pi(S_{i-1}) + k \cdot y_i(v_i) \\
&< \pi(S_{i-1} + x_{i-1}) = OPT_L
\end{aligned}$$

which contradicts the definition of OPT_L.

Recall that, in each iteration of *MCGreedy*, it add into S_{i-1} the node v that leads to the largest $\xi(S_{i-1} \cup \{v\})$. Therefore,

$$\xi(S_i) - \xi(S_{i-1}) \geq \xi(S_i \cup \{v\}) - \xi(S_{i-1}) \qquad (16)$$

Combining Equations 15 and 16, we have

$$\begin{aligned}
x_{i-1} - x_i &= \pi(S_i) - \pi(S_{i-1}) \\
&\geq \xi(S_i) - \frac{\epsilon}{2k} OPT_L - \xi(S_{i-1}) + \big(\xi(S_{i-1}) - \pi(S_{i-1})\big) \\
&\geq \xi(S_{i-1} \cup \{v_i\}) - \xi(S_{i-1}) \qquad (17) \\
&\quad - \frac{\epsilon}{2k} OPT_L + \big(\xi(S_{i-1}) - \pi(S_{i-1})\big) \\
&\geq \pi\big(S_{i-1} \cup \{v_i\}\big) - \pi(S_{i-1}) - \frac{\epsilon}{k} OPT_L \\
&\geq \frac{1}{k} x_{i-1} - \frac{\epsilon}{k} OPT_L \qquad (18)
\end{aligned}$$

Equation 18 leads to

$$\begin{aligned}
x_k &\leq \big(1 - \frac{1}{k}\big) \cdot x_{k-1} + \frac{\epsilon}{k} OPT_L \\
&\leq \big(1 - \frac{1}{k}\big)^2 \cdot x_{k-2} + \big(1 + \big(1 - \frac{1}{k}\big)\big) \cdot \frac{\epsilon}{k} OPT_L \\
&\leq \big(1 - \frac{1}{k}\big)^k \cdot x_0 + \sum_{i=0}^{k-1} \big(\big(1 - \frac{1}{k}\big)^i \cdot \frac{\epsilon}{k} OPT_L\big) \\
&= \big(1 - \frac{1}{k}\big)^k \cdot OPT_L + \big(1 - \big(1 - \frac{1}{k}\big)^k\big) \cdot \epsilon \cdot OPT_L \\
&\leq \frac{1}{e} \cdot OPT_L - \big(1 - \frac{1}{e}\big) \cdot \epsilon \cdot OPT_L
\end{aligned}$$

Therefore,

$$\begin{aligned}
\pi(S_k) &= OPT_L - x_k \\
&\leq (1 - 1/e) \cdot (1 - \epsilon) \cdot OPT_L \\
&\leq (1 - 1/e - \epsilon) \cdot OPT_L
\end{aligned}$$

Thus, the lemma is proved. □

Assume that we know OPT_L in advance and set r to the smallest value satisfying the above inequality, in *MCGreedy*'s favour. In that case, the time complexity of *MCGreedy* is $O(k^3 lmn^2 \epsilon^{-2} \log n / OPT_L)$. Towards comparing *MCGreedy* to *RPS*, we show the following upper bound on the value of γ.

CLAIM 1. $\gamma \leq \frac{n}{n+1}$

Proof. Let M_R be the random instance of $R_g(A_C)$ used to compute R. Furthermore, let us assume that $|M_R| \geq 2$ so that at least one non-seed node is influenced by campaign C. Then, from Lemma 4 and the definition of γ we have

$$\frac{1}{\gamma} = 1 + \frac{m}{n} \cdot \frac{\mathbb{E}[\pi(\{v^*\})]}{\mathbb{E}[\omega_C(R)]}$$

Then, observe that the expected number of nodes saved by v^* is at least $\Pr[v^* \in M_R]$. That is, if $v^* \in M_R$ then campaign L can save at least one node, namely v^* itself. Giving

$$\Pr[v^* \in M_R] = \sum_{v \in M_R} \frac{deg(v)}{m} \geq \sum_{v \in M_R} \frac{1}{m} = \frac{|M_R|}{m}$$

Therefore, $\frac{\mathbb{E}[\pi(\{v^*\})]}{\mathbb{E}[\omega_C(R)]} \geq \frac{1}{m}$. Then, we have $\frac{1}{\gamma} \geq 1 + \frac{m}{n} \cdot \frac{1}{m} = 1 + \frac{1}{n}$. Thus, we get $\gamma \leq \frac{n}{n+1}$ proving the claim. □

Claim 1 shows that the expected runtime for *RPS* is at most $O((k+l)mn\epsilon^{-2}\log n)$. As a consequence, given that $OPT_L \leq n$, the expected runtime of *MCGreedy* is substantially larger than the expected runtime of *RPS*. In practice, we observe that for typical social networks $OPT_L \ll n$ and $\frac{1}{1-\gamma} \ll n+1$ resulting in superior scalability of *RPS* compared to *MCGreedy*. Table 3 confirms that $\frac{1}{1-\gamma} \ll n+1$ on our small datasets.

A Lower Bound for EIL In the theorem below, we provide a lower bound on the time it takes for any algorithm to compute a β-approximation for the EIL problem given uniform node sampling and an adjacency list representation. Thus, we rule out the possibility of a sublinear time algorithm for the EIL problem for an arbitrary β.

Theorem 4. Let $0 < \epsilon < \frac{1}{10e}$, $\beta \leq 1$ be given. Any randomized algorithm for EIL that returns a set of seed nodes with approximation ratio β, with probability at least $1 - \frac{1}{e} - \epsilon$, must have a runtime of at least $\frac{\beta(m+n)}{24\min\{k,1/\beta\}}$.

Proof. We make use of Yao's Minmax Lemma for the performance of Las Vegas (LV) randomized algorithms on a family of inputs [69]. Precisely, the lemma states that the least expected cost of deterministic LV algorithms on a distribution over a family of inputs lower bounds the expected cost of the optimal randomized LV algorithm over that family of inputs. We build such an input family of lower bound graphs via the use of a novel gadget.

Throughout the proof we assume all edge probabilities for both campaigns are 1. Note, for a graph consisting of $p = n/2$ connected pairs for which each pair contains a node $u \in A_C$, an algorithm must return at least βk nodes to obtain an approximation ratio of β. Doing so in at most $\beta^2 n/2$ queries requires that $2\beta k \leq \beta^2 n$, which implies $2k/\beta \leq n$. We can therefore assume $2k/\beta \leq n$.

Define the cost of the algorithm as 0 if it returns a set of seed nodes with approximation ratio better than β and 1 otherwise. As the cost of an algorithm equals its probability of failure, we can think of it as a LV algorithm. Assume for notational simplicity that $\beta = 1/T$ where T is an integer. We will build a family of lower bound graphs, one for each value of n (beginning from $n = 1+T$); each graph will have $m \leq n$, so it will suffice to demonstrate a lower bound of $\frac{n}{12T\min\{k,T\}}$.

We now consider the behaviour of a deterministic algorithm A with respect to the uniform distribution on the constructed family of inputs. For a given value T the graph would be made from k components of size $2T$ and $p = \frac{n-2kT}{2}$ connected pairs (recall that $2kT = 2k/\beta \leq n$). Specifically, the k components of size $2T$ are structured as follows: for each component there is a *hub* node v_h that is connected to $2T-2$ leaf nodes and a node $u \in A_C$. Furthermore, each of the p pairs also contains one node $u \in A_C$. If algorithm A returns seed nodes from l of the k components of size $2T$, it achieves a total prevention of $l \cdot (2T-1) + (k-l)$ since choosing either the hub node v_h or any leaf node will result in saving all $2T-1$ eligible nodes in the component. Thus, to attain an approximation factor better than $\beta = \frac{1}{T}$, we must have $l \cdot (2T-1) + (k-l) \geq \frac{1}{T} \cdot k \cdot (2T-1)$, which implies $l \geq \frac{k}{2T}$ for any $T > 1$.

Suppose $k > 12T$. The condition $l \geq \frac{k}{2T}$ implies that at least $\frac{k}{2T}$ of the large components must be queried by the algorithm, where each random query has probability $\frac{k\cdot(2T-1)}{n-(p+k)} \geq \frac{kT}{n}$ of hitting a large component. If the algorithm makes fewer than $\frac{n}{6T^2}$ queries, then the expected number of components hit is $\frac{n}{6T^2} \cdot \frac{kT}{n} = \frac{k}{6T}$. Chernoff bounds then imply that the probability of hitting more than $\frac{k}{2T}$ components is no more than $e^{-\frac{k}{6T}\cdot 2/3} \leq \frac{1}{e^{4/3}} < 1 - \frac{1}{e} - \epsilon$, a contradiction.

If $k \leq 12T$ then we need that $l \geq 1$, which occurs only if the algorithm queries at least one of the $k \cdot (2T-1)$ vertices in the large components. With $\frac{n}{k\cdot(2T-1)}$ queries, for n large enough, this happens with probability less than $\frac{1}{e} - \epsilon$, a contradiction.

We conclude that, in all cases, at least $\frac{n}{24T\min\{k,T\}}$ queries are necessary to obtain an approximation factor better than $\beta = \frac{1}{T}$ with probability at least $1 - \frac{1}{e} - \epsilon$, as required. By Yao's Minmax Principle this gives a lower bound of $\Omega(\frac{n}{24T\min\{k,T\}})$ on the expected performance of any randomized algorithm, on at least one of the inputs.

Finally, the construction can be modified to apply to non-sparse networks. For any $d \leq n$, we can augment our graph by overlaying a d-regular graph with exponentially small weight on each edge. This does not significantly impact the prevention of any set, but increases the time to decide if a node is in a large component by a factor of $O(d)$ (as edges must be traversed until one with non-exponentially-small weight is found). Thus, for each $d \leq n$, we have a lower bound of $\Omega(\frac{nd}{24T\min\{k,T\}})$ on the expected performance of A on a distribution of networks with $m = nd$ edges. □

7. GENERALIZATION TO THE MULTI-CAMPAIGN TRIGGERING MODEL

The *triggering model* is an influence propagation model that generalizes the IC and LT models. It assumes that each node v is associated with a triggering distribution $\mathcal{T}(v)$ over the power set of v's incoming neighbors. An influence propagation process under the triggering model works as follows: (1) for each node v, take a sample from $\mathcal{T}(v)$ and define the sample as the triggering set of v, then (2) at timestep 1 activate the seed set S, and (3) in subsequent timesteps, if an

active node appears in the triggering set of v, then v becomes active. The propagation terminates when no more nodes can be activated.

We can define a *multi-campaign* version of the triggering model (MCT) that generalizes the MCIC model by associating each node with a *campaign-specific* triggering distribution $\mathcal{T}_Z(v)$ where $Z \in \{C, L\}$. The propagation process under MCT proceeds exactly as under MCIC with the exception that activation between rounds (step 2) is determined by $\mathcal{T}_C(v)$ and $\mathcal{T}_L(v)$. To the best of our knowledge, we are the first to formally define a multi-campaign version of the triggering model.

The key aspect of the MCIC model that enabled the existence of obstructed nodes is that the two campaigns are allowed to propagate along *different sets of edges* in a possible world X. This is exactly the intuition captured by the example in Figure 1 and is caused by L and C having separate propagation probabilities in $\mathcal{G}$. As a result, the campaigns traverse potentially unique graphs in X and results in the possibility of the obstruction of L by C. This observation holds under the more general setting of MCT due to the campaign-specific triggering sets and so the obstruction phenomenon exists under the MCT model.

REMARK 1. Propagation under the multi-campaign triggering model requires the notion of obstruction to correctly characterize the conditions required to save an arbitrary node.

Following the observations made in [10], our solutions can be easily extended to operate under the multi-campaign triggering (MCT) model with a modified high effectiveness property. Under MCT, the high effectiveness property asserts that $\mathcal{T}_L(v) = in(v)$ where $in(v)$ is the set of in-neighbours of v in G. Observe that Algorithm 1 does not rely on anything specific to the MCIC model, except a subroutine to generate random RRC sets. Thus, we can revise the definition of RRC sets to accommodate the MCT model.

Suppose that we generate a possible world g from G for C by sampling a node set T for each node v from its triggering distribution $\mathcal{T}_C(v)$ and removing any outgoing edge of v that does not point to a node in T. Let $\mathcal{G}$ be the distribution of g induced by the random choices of triggering sets. We refer to $\mathcal{G}$ as the *triggering graph distribution* for campaign C in G. Similar to the MCIC setting, a possible world for L under the high effectiveness property is the whole graph G. For any given node v and a possible world g for C sampled from $\mathcal{G}$, we define the reverse reachable without cutoff (RRC) set for v in g as the set of nodes that can save v in g. In addition, we define a random RRC set as one that is generated on an instance of g randomly sampled from $\mathcal{G}$, for a node selected from g uniformly at random. These random RRC sets are constructed in the same fashion as Algorithm 2 with the modification that the forward randomized BFS samples triggering sets to determine which edges to traverse. By incorporating such an updated forward BFS into Algorithm 1, our solution extends to the multi-campaign triggering model.

Random RRC sets, as defined above, are constructed by a randomized BFS as follows. Let v be a randomly selected node. Given v, we first take a sample T from v's triggering distribution $\mathcal{T}_C(v)$, and then put all nodes in T into a queue. Then, we iteratively extract the node at the top of the queue; for each node u extracted, we sample a set T' from u's triggering distribution, and we insert any unvisited node in T' into the queue. When the queue becomes empty, we terminate the process, and form a random RRC set with the nodes visited during the process. The expected cost of the whole process is $O(EPT)$, where EPT denotes the expected number of edges in G that point to the nodes in a random RRC set. This expected time complexity is the same as Algorithm 2 for generating random RRC sets under the MCIC model.

Our next step is to show that the revised solution retains the performance guarantees of *RPS*. We first present an extended version of Lemma 2 for the MCT model. (The proof of the lemma is almost identical to that of Lemma 2.)

LEMMA 10. Let S be a fixed set of nodes, v be a fixed node, and $\mathcal{G}$ be the triggering graph distribution for G. Suppose that we generate an RRC set R for v on a graph g sampled from $\mathcal{G}$. Let ρ_1 be the probability that S overlaps with R, and ρ_2 be the probability that S (as a seed set for campaign L) can save v in an prevention process on G under the MCT model. Then, $\rho_1 = \rho_2$.

Proof. Let S be a fixed set of nodes, and v be a fixed node. Suppose that we generate an RRC set R for v on a graph $g \sim \mathcal{G}$. Let ρ_1 be the probability that S overlaps with R and let ρ_2 be the probability that S, when used as a seed set, can save v in a prevention process on $\mathcal{G}$. By the definition of RRC sets under the MCT model, if $v \in R_g(A_C)$ then ρ_1 equals the probability that a node $u \in S$ is a saviour for v. That is, ρ_1 equals the probability that G contains a directed path from $u \in S$ to v and $u \notin \tau(v)$ and 0 if $v \notin R_g(A_C)$. Meanwhile, if $v \in R_g(A_C)$ then ρ_2 equals the probability that a node $u \in S$ can save v (i.e. $v \in (cl_g(u) \setminus obs_g(u))$) and 0 if $v \notin R_g(A_C)$. It follows that $\rho_1 = \rho_2$ due to the equivalence between the set of saviours for v and the ability to save v. □

We note that all of the theoretical analysis of *RPS* is based on the Chernoff bounds and Lemma 2, without relying on any other results specific to the MCIC model. Therefore, once we establish an equivalent to Lemma 2 (Lemma 10), it is straightforward to combine it with the Chernoff bounds to show that, under the MCT model, *RPS* provides the same performance guarantees as in the case of the MCIC model. Thus, we have the following theorem:

THEOREM 5. Under MCT, *RPS* runs in $O((k+l)(m+n)(1/(1-\gamma))\log n/\epsilon^2)$ expected time, and returns a

TABLE 2: Dataset Statistics

Name	$\|V\|$	$\|E\|$	Average degree
nethept	15,229	62,752	4.1
word_assoc	10,617	72,172	6.8
dblp-2010	326,186	1,615,400	6.1
cnr-2000	325,557	3,216,152	9.9
ljournal-2008	5,363,260	79,023,142	28.5

TABLE 3: $\frac{1}{1-\gamma}$ values for small datasets.

	word_assoc		nethept	
k	top1	top5	top1	top5
1	23.4471	25.9804	48.3194	48.619
10	24.8521	26.6518	60.5	43.1875
20	24.7509	25.2607	57.0111	61.4167
max	10,618	10,618	15,230	15,230

$(1-1/e-\epsilon)$-approximate solution with at least $1-4n^{-l}$ probability.

8. EXPERIMENTS

The primary goal of this work was to scale up misinformation mitigation while still providing approximation guarantees. From this perspective, there is only one existing approach, namely *MCGreedy*, which satisfies the requirement of providing a solution to the *EIL* problem with guarantees on the solution quality. We make a thorough comparison of *RPS* with *MCGreedy* in this section.

Notably, we made a conscious choice not to compare *RPS* to the heuristic baselines suggested in [7] for the following two reasons. First, since *RPS* is equivalent to *MCGreedy* with respect to how it derives a solution (i.e. greedily optimizing an unbiased estimator), the solution quality of these two approaches are identical. The experiments in [7] confirm that *MCGreedy* strongly outperforms the heuristic baselines with respect to solution quality. Therefore, *RPS* will outperform all the other heuristic baseline methods, which do not come with approximation guarantees, when comparing solution quality. Second, since the heuristics included as baseline methods in [7] are based on simple structural properties of the network, their solutions can be computed in time linear in the number of nodes and edges in the network. On the other hand, *RPS* takes time $O((k+l)(n+m)(\frac{1}{1-\gamma})\log n/\epsilon^2) \approx O((m+n)\log n)$ due to its emphasis on solution quality. For these reasons, we believe a comparison with the various heuristics from [7] would not provide meaningful results with respect to running time.

Focusing on the *algorithm efficiency*, measured in runtime, we demonstrate that *RPS* provides a significant improvement of several orders of magnitude over *MCGreedy*. Further, we confirm that $\frac{1}{1-\gamma} \ll n+1$ on our small datasets which is strong evidence that *RPS* will outperform *MCGreedy* on typical social networks. Finally, we observe that the vast majority of the computation time is spent on generating the RRC sets for $\mathcal{R}$.

All of our algorithms are implemented in C++ (available at `https://github.com/stamps`) and tested on a machine with dual 6 core 2.10GHz Intel Xeon CPUs, 128GB RAM and running Ubuntu 14.04.2.

Datasets. The network statistics for all of the datasets we consider are shown in Table 2. We obtained the datasets from Laboratory of Web Algorithmics.[4] We divide the datasets by horizontal lines according to their size, small (S), medium (M), and large (L).

Propagation Model. We consider the MCIC model (see Section 2.1) of Budak et al. We set the propagation probability of each edge e as follows: we first identify the node v that e points to, and then set $p(e) = 1/i$, where i denotes the in-degree of v. This setting of $p(e)$ is widely adopted in prior work [15, 70, 12, 13].

Parameters. Unless otherwise specified, we set $\epsilon = 0.1$ in our experiments. We set l in a way that ensures a success probability of $1-1/n$. For *MCGreedy*, we set the number of Monte Carlo steps to $r = 10000$, following the standard practice in the literature. Note that this choice of r is to the advantage of *MCGreedy* because the value of r required to achieve the same theoretical guarantees as *RPS* in our experiments is always much larger than 10000. In each experiment, we repeat each run five times and report the average result.

We are interested in simulating the misinformation prevention process when the bad campaign C has a sizable influence on the network to best demonstrate how the techniques could be used in real world settings. That is, we believe the scenario in which we are attempting to prevent the spread of misinformation when the bad campaign has the ability to influence a large fraction of the network to be more relevant than when only a very small number of users would adopt the bad campaign. Towards this end, we first compute the most influential vertices for each network and then randomly select a misinformation seed ($|A_C| = 1$) from the 99th (*top-1*) and 95th (*top-5*) percentiles for each experiment. This process ensures the misinformation has a large potential influence in the network.

The focus of our experiments is *algorithm efficiency* measured in runtime where our goal is to demonstrate the superior performance of *RPS* compared to *MCGreedy*. Meanwhile, we observed that the *algorithm accuracy* (measured in percentage of nodes saved) of *RPS* matches *MCGreedy* very closely. We observe that, consistent with the results reported in [7], *RPS* quickly approaches a maximal expected prevention value as k increases across all datasets. This is natural since both *RPS* and *MCGreedy* are maximizing a submodular

[4] `http://law.di.unimi.it/datasets.php`

objective function in a greedy fashion. The novelty of *RPS* addresses the scalability hurdle in a similar sense to Borgs et. al. [9] in relation to Kempe et. al. [8]. For a detailed comparison of the accuracy of *MCGreedy* compared to a number of natural heuristics we refer the interested reader to [7].

Plots. First, we plot the runtimes of *MCGreedy* and *RPS* for a single seed in Figure 3 and observe that *RPS* provides a significant improvement of several orders of magnitude over *MCGreedy*. Note, we only compare *RPS* to *MCGreedy* on the smallest networks due to the substantial runtime required for *MCGreedy*. Furthermore, for similar scaling issues of *MCGreedy*, we restrict our comparison to $k = 1$. However, since both approaches scale linearly with k we can conclude that *RPS* offers a tremendous runtime improvement over the approach of [7].

In Figure 4 we plot the prevention (average number of nodes saved) achieved by *MCGreedy* and *RPS* on the two small datasets for $k = 1$. The blue (red) bars correspond to when the A_C is selected uniformly at random from the *top-1* (*top-5*) set. Note, since the selection is randomized, it is possible that the misinformation seed generated from the *top-5* set has higher total influence than when selecting from the *top-1* set. In fact, we see this in Figure 4 for the nethept dataset, where the total influence achieved when A_C was selected from the *top-5* set is larger than when selected from the *top-1* set, and thus there is a greater opportunity to save nodes from adopting the misinformation. Across both datasets and A_C selection methods, we observe that *RPS* achieves comparable prevention to *MCGreedy* up to sampling errors.

Next, we show the total runtimes (Figures 5, 6) and a computation breakdown for the medium datasets (Figure 7). We observe that the vast majority of the computation time is spent on generating the RRC sets for $\mathcal{R}$. Furthermore, the amount of time spent on computing a lower bound increases across all datasets though remains a small fraction of the overall runtime. As expected, the time spent refining the lower bound estimate remains a very small fraction of total computation time due to the small number of iterations of the algorithm that improves the lower bound estimate and only takes up a relatively large fraction of the computation time on the cnr-2000 top5 dataset (Figure 7c). The density of the cnr-2000 network leads to larger RRC sets that results in a larger fraction of time spent on computing and refining a lower bound. We observe similar trends on the small and large datasets where the generation of RRC sets dominates the runtime breakdown by an even larger margin. Finally, we plot the memory consumption statistics in Figure 8.

Running-time Results. We compare the runtime trends of our results for the EIL problem to those of [10] for the IM problem. Tang et al. report that, when k increases, the runtime of their approach (*TIM*) tends to decrease before eventually increasing. They explain this by considering the breakdown of the computation times required by each algorithm in *TIM*. They observe that the computation time is mainly incurred by their analog to Algorithm 1 (the node selection phase) which is primarily determined by the number θ of RR sets that need to be generated. They have $\theta = \lambda / KPT^+$, where λ is analogous to ours, and KPT^+ is a lower-bound on the optimal influence of a size-k node set. In both the IC and MCIC models, the analogs of λ and KPT^+ increase with k, and it happens that for the IM problem, Tang et al. observe that the increase of KPT^+ is more pronounced than that of λ for smaller values of k, which leads to the decrease in *TIM*'s runtime.

On the contrary, for the EIL problem, the increase of KPT^+ does not dominate to a point that the runtime of *RPS* decreases as k increases. Instead, we see a linear increase in runtime as k increases for all the networks considered. To explain, consider how KPT^+ grows in each setting. In the MCIC model we see that KPT^+ rapidly approaches its maximal value which corresponds to the growth of KPT^+ plateauing much sooner. In contrast, in the IC model, the analogous KPT^+ value continues to grow at a significant rate for a wider range of k values since the ceiling for the maximal influence is not tied to a second campaign, as it is in the MCIC model. As such, the influence estimates do not level off as quickly. This translates to the growth of KPT^+ outpacing the growth of λ.

Memory Consumption. Another set of experiments monitors the memory consumption required to store the RRC set structure $\mathcal{R}$. We observe that the size of $\mathcal{R}$ for the EIL problem is larger than that required by the IM problem. Using the "hypergraph" nomenclature due to Borgs et al. $\mathcal{R}$ is viewed as a hypergraph with each RRC set in $\mathcal{R}$ corresponding to a hyperedge. We observe that the hyperedges generated for the IM problem are non-empty in every iteration of the algorithm. Additionally, each hyperedge has relatively small size. The result is that the hypergraph generated for the IM problem is very dense, but each hyperedge is relatively "light" (i.e. it contains few nodes).

In contrast, in each iteration of *RPS* we have a substantial probability to produce an empty RRC set, since we require that a randomly selected node is in the randomized BFS tree resulting from the influence propagation process initialized at A_C. These empty RRC sets are necessary for the computation of expected prevention to be accurate, but results in a hypergraph that differs significantly in structure from those of the IM problem.

In particular, since the *generateRRC* algorithm is a deterministic BFS (with specialized stopping conditions to account for cutoff nodes) it reaches a much larger fraction of the network. Therefore, while there are far fewer non-empty hyperedges generated, they are much large in size: often on the order of half the network.

Thus, the resulting hypergraph is sparse, but contains very "heavy" edges. These two opposing metrics, a dense hypergraph with "light" hyperedges versus a sparse hypergraph with "heavy" hyperedges, result in the latter requiring more memory to store. Despite a larger memory requirement compared to the single campaign setting we show that our approach has the ability to scale far beyond what was achieved by Budak et al. and provides orders of magnitude improvement for the runtime.

9. CONCLUSION & FUTURE WORK

In this work we presented *RPS*, a novel and scalable approach to the EIL problem. We showed the correctness and a detailed running-time analysis of our approach. Furthermore, we provided two lower bound results: one on the running-time requirement for any approach to solve the EIL problem and another on the number of Monte Carlo simulations required by *MCGreedy* to return a correct solution with high probability. As a result, the expected runtime of *RPS* is always less than the expected runtime of *MCGreedy*. Finally, we describe how our approach can be generalized to a multi-campaign triggering model. In future work we plan to investigate how to adapt our approach to a scenario where the source of the misinformation is only partially known.

10. DATA AVAILABILITY

The source code underlying this article are available at `github.com/stamps/misinformation_prevention`. The datasets were derived from sources in the public domain: Laboratory of Web Algorithmics at `http://law.di.unimi.it/datasets.php`.

REFERENCES

[1] Foster, P. (2018). 'bogus' ap tweet about explosion at the white house wipes billions off us markets.

[2] Oppenheim, M. (2018). Youtube shooting: Twitter and facebook explodes with misinformation and hoaxes.

[3] Graham, C. (2018). Youtube employee's twitter account hacked to spread fake news during attack.

[4] Hautala, L. (2018). Reddit was a misinformation hotspot in 2016 election, study says.

[5] Solon, O. (2018). Facebook's failure: did fake news and polarized politics get trump elected?

[6] Abeshouse, B. (2018). Troll factories, bots and fake news: Inside the wild west of social media.

[7] Budak, C., Agrawal, D., and El Abbadi, A. (2011) Limiting the spread of misinformation in social networks. *Proceedings of the 20th International Conference on World Wide Web*, New York, NY, USA WWW '11 665–674. Association for Computing Machinery.

[8] Kempe, D., Kleinberg, J., and Tardos, E. (2003) Maximizing the spread of influence through a social network. *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA KDD '03 137–146. Association for Computing Machinery.

[9] Borgs, C., Brautbar, M., Chayes, J., and Lucier, B. (2014) Maximizing social influence in nearly optimal time. *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pp. 946–957. SIAM.

[10] Tang, Y., Xiao, X., and Shi, Y. (2014) Influence maximization: Near-optimal time complexity meets practical efficiency. *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, New York, NY, USA SIGMOD '14 75–86. Association for Computing Machinery.

[11] Tang, Y., Shi, Y., and Xiao, X. (2015) Influence maximization in near-linear time: A martingale approach. *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, New York, NY, USA SIGMOD '15 1539–1554. Association for Computing Machinery.

[12] Jung, K., Heo, W., and Chen, W. (2012) Irie: Scalable and robust influence maximization in social networks. *ICDM '12*, pp. 918–923. IEEE.

[13] Wang, C., Chen, W., and Wang, Y. (2012) Scalable influence maximization for independent cascade model in large-scale social networks. *Data Mining and Knowledge Discovery*, **25**, 545–576.

[14] Leskovec, J., Huttenlocher, D., and Kleinberg, J. (2010) Predicting positive and negative links in online social networks. *Proceedings of the 19th International Conference on World Wide Web*, New York, NY, USA WWW '10 641–650. Association for Computing Machinery.

[15] Chen, W., Yuan, Y., and Zhang, L. (2010) Scalable influence maximization in social networks under the linear threshold model. *ICDM '10*. IEEE.

[16] Goyal, A., Bonchi, F., Lakshmanan, L. V. S., and Venkatasubramanian, S. (2013) On minimizing budget and time in influence propagation over social networks. *Social Netw. Analys. Mining*, **3**, 179–192.

[17] Nguyen, H. T., Thai, M. T., and Dinh, T. N. (2016) Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks. *Proceedings of the 2016 International Conference on Management of Data*, pp. 695–710.

[18] Huang, K., Wang, S., Bevilacqua, G., Xiao, X., and Lakshmanan, L. V. (2017) Revisiting the stop-and-stare algorithms for influence maximization. *Proceedings of the VLDB Endowment*, **10**, 913–924.

[19] Tang, J., Tang, X., Xiao, X., and Yuan, J. (2018) Online processing algorithms for influence maximization. *Proceedings of the 2018 International Conference on Management of Data*, pp. 991–1005.

[20] Bharathi, S., Kempe, D., and Salek, M. (2007) Competitive influence maximization in social networks. *International workshop on web and internet economics*, pp. 306–311. Springer.

[21] Lin, Y. and Lui, J. C. (2015) Analyzing competitive influence maximization problems with partial information: An approximation algorithmic framework. *Performance Evaluation*, **91**, 187–204.

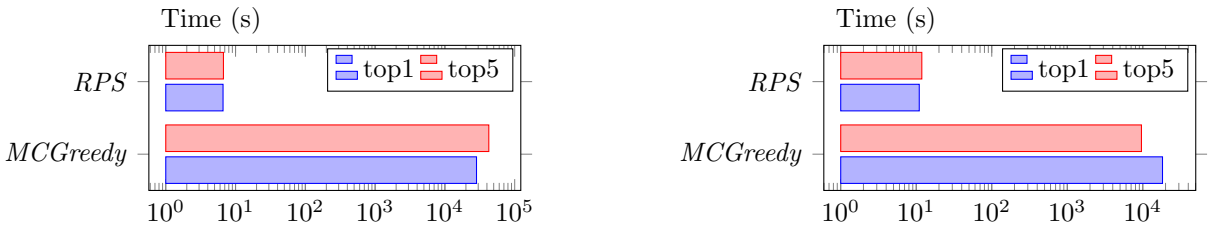

FIGURE 3: Runtimes comparison between *RPS* and *MCGreedy* for wordassociation-2011 (left) and nethept (right) datasets.

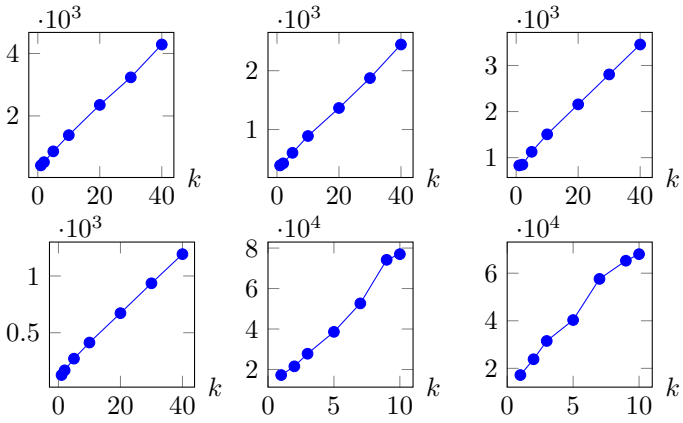

FIGURE 6: Runtimes (s) for medium & large datasets. Listed left to right: dblp top1, dblp top5, cnr top1, cnr top5, ljournal-2008 top1, ljournal-2008 top5.

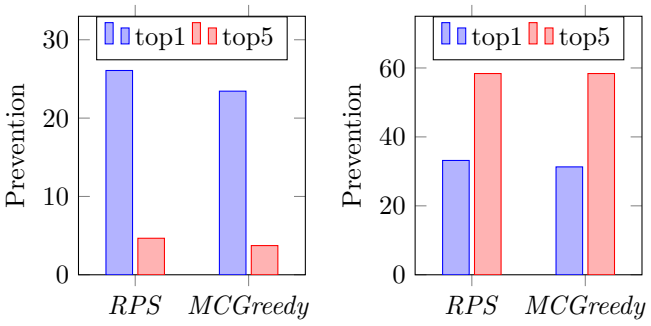

FIGURE 4: Prevention comparison between *RPS* and *MCGreedy* for wordassociation-2011 (left) and nethept (right) datasets.

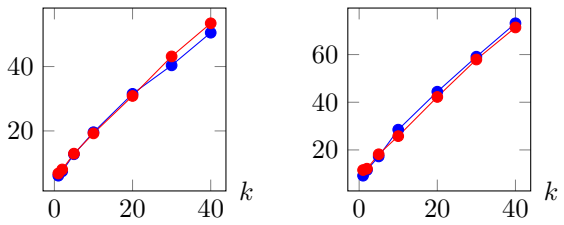

FIGURE 5: Runtimes (s) for small datasets. word_assoc on the left and nethept on the right with blue for top1 and red for top5.

[22] Pathak, N., Banerjee, A., and Srivastava, J. (2010) A generalized linear threshold model for multiple cascades. *ICDM '10*, pp. 965–970. IEEE.

[23] Li, Y., Chen, W., Wang, Y., and Zhang, Z.-L. (2013) Influence diffusion dynamics and influence maximization in social networks with friend and foe relationships. *WSDM '13*, pp. 657–666. ACM.

[24] He, X., Song, G., Chen, W., and Jiang, Q. (2012) Influence blocking maximization in social networks under the competitive linear threshold model. *SDM '12*, pp. 463–474. SIAM.

[25] Fan, L., Lu, Z., Wu, W., Thuraisingham, B., Ma, H., and Bi, Y. (2013) Least cost rumor blocking in social networks. *ICDCS '13*, pp. 540–549. IEEE.

[26] Song, C., Hsu, W., and Lee, M. L. (2017) Temporal influence blocking: Minimizing the effect of misinformation in social networks. *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, pp. 847–858. IEEE.

[27] Tong, A., Du, D.-Z., and Wu, W. (2018) On misinformation containment in online social networks. *Advances in neural information processing systems*, pp. 341–351.

[28] Tong, G. A. and Du, D.-Z. (2019) Beyond uniform reverse sampling: A hybrid sampling technique for misinformation prevention. *IEEE INFOCOM 2019-IEEE conference on computer communications*, pp. 1711–1719. IEEE.

[29] Tong, G., Wu, W., Guo, L., Li, D., Liu, C., Liu, B., and Du, D.-Z. (2017) An efficient randomized algorithm for rumor blocking in online social networks. *IEEE Transactions on Network Science and Engineering*, **7**, 845–854.

[30] Saxena, A., Hsu, W., Lee, M. L., Leong Chieu, H., Ng, L., and Teow, L. N. (2020) Mitigating misinformation in online social network with top-k debunkers and evolving user opinions. *Companion Proceedings of the Web Conference 2020*, pp. 363–370.

[31] Pham, C. V., Phu, Q. V., and Hoang, H. X. (2018) Targeted misinformation blocking on online social networks. *Asian Conference on Intelligent Information and Database Systems*, pp. 107–116. Springer.

[32] Pham, C. V., Phu, Q. V., Hoang, H. X., Pei, J., and Thai, M. T. (2019) Minimum budget for misinformation blocking in online social networks. *Journal of Combinatorial Optimization*, **38**, 1101–1127.

[33] Fang, Q., Chen, X., Nong, Q., Zhang, Z., Cao, Y., Feng, Y., Sun, T., Gong, S., and Du, D.-Z. (2018) General rumor blocking: An efficient random algorithm with martingale approach. *International Conference on Algorithmic Applications in Management*, pp. 161–176. Springer.

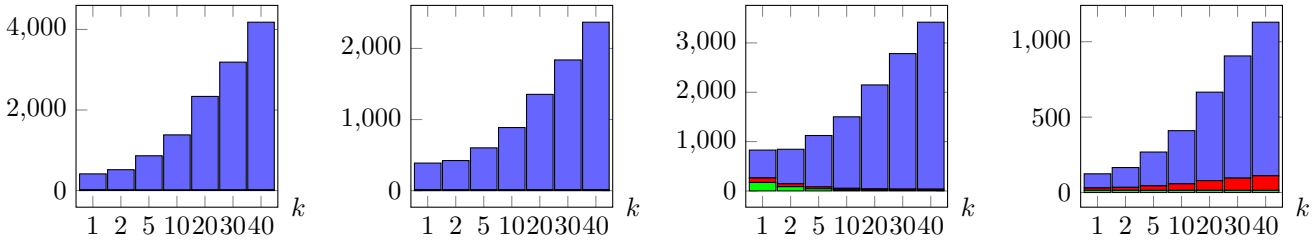

FIGURE 7: Breakdown of computation time (s) for medium datasets. Blue stack corresponds to Algorithm 1, red to improving the lower bound estimation, and green (which is almost invisible) to computing the initial lower bound estimate. Listed left to right: dblp top1, dblp top5, cnr top1, cnr top5.

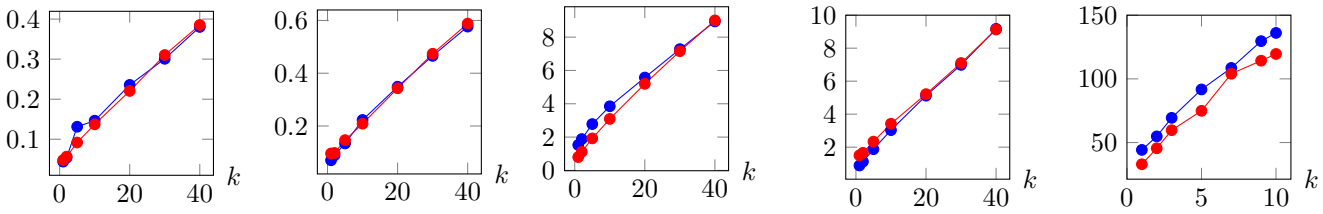

FIGURE 8: Memory consumption (Gb) for all datasets. Blue corresponds to top1 and red to top5. Listed left to right: word_assoc, nethept, cnr, dblp & ljournal-2008.

[34] Prakash, B. A., Chakrabarti, D., Valler, N. C., Faloutsos, M., and Faloutsos, C. (2012) Threshold conditions for arbitrary cascade models on arbitrary networks. *Knowledge and information systems*, **33**, 549–575.

[35] Prakash, B. A., Adamic, L., Iwashyna, T., Tong, H., and Faloutsos, C. (2013) Fractional immunization in networks. *Proceedings of the 2013 SIAM International Conference on Data Mining*, pp. 659–667. SIAM.

[36] Zhang, Y. and Prakash, B. A. (2014) Dava: Distributing vaccines over networks under prior information. *Proceedings of the 2014 SIAM International Conference on Data Mining*, pp. 46–54. SIAM.

[37] Simpson, M., Srinivasan, V., and Thomo, A. (2016) Clearing contamination in large networks. *IEEE Transactions on Knowledge and Data Engineering*, **28**, 1435–1448.

[38] Tong, H., Prakash, B. A., Eliassi-Rad, T., Faloutsos, M., and Faloutsos, C. (2012) Gelling, and melting, large graphs by edge manipulation. *CIKM*, pp. 245–254.

[39] Medya, S., da Silva, A. L., and Singh, A. K. (2019) Influence minimization under budget and matroid constraints: Extended version. *ArXiv*, **abs/1901.02156**.

[40] Khalil, E. B., Dilkina, B., and Song, L. (2014) Scalable diffusion-aware optimization of network topology. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1226–1235.

[41] Chen, C., Tong, H., Prakash, B. A., Tsourakakis, C. E., Eliassi-Rad, T., Faloutsos, C., and Chau, D. H. (2015) Node immunization on large graphs: Theory and algorithms. *TKDE*, **28**, 113–126.

[42] Zhang, Y., Ramanathan, A., Vullikanti, A., Pullum, L., and Prakash, B. A. (2019) Data-driven efficient network and surveillance-based immunization. *Knowledge and Information Systems*, **61**, 1667–1693.

[43] Facebook (2019). How is facebook addressing false news? `https://www.facebook.com/help/1952307158131536`, Last accessed on 2020-07-07.

[44] Facebook (2019). Helping to protect the 2020 us elections. `https://about.fb.com/news/2019/10/update-on-election-integrity-efforts/`, Last accessed on 2020-07-07.

[45] Twitter (2020). Notices on twitter and what they mean. `https://help.twitter.com/en/rules-and-policies/notices-on-twitter`, Last accessed on 2020-07-07.

[46] Twitter (2020). Our range of enforcement options. `https://help.twitter.com/en/rules-and-policies/enforcement-options`, Last accessed on 2020-07-07.

[47] Instagram (2019). Instagram adds 'false information' labels to prevent fake news from going viral. `https://me.mashable.com/tech/7586/instagram-adds-false-information-labels-to-prevent-fake-news-from-going-viral`, Last accessed on 2020-07-07.

[48] Pinterest (2019). Health misinformation. `https://help.pinterest.com/en/article/health-misinformation`, Last accessed on 2020-07-07.

[49] Shu, K., Sliva, A., Wang, S., Tang, J., and Liu, H. (2017) Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explorations Newsletter*, **19**, 22–36.

[50] Zhou, D., He, J., Yang, H., and Fan, W. (2018) Sparc: Self-paced network representation for few-shot rare category characterization. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2807–2816.

[51] Zhou, D., Zhang, S., Yildirim, M. Y., Alcorn, S., Tong, H., Davulcu, H., and He, J. (2017) A local algorithm for structure-preserving graph cut. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 655–664.

[52] Hassan, N., Zhang, G., Arslan, F., Caraballo, J., Jimenez, D., Gawsane, S., Hasan, S., Joseph, M., Kulkarni, A., Nayak, A. K., et al. (2017) Claimbuster: The first-ever end-to-end fact-checking system. *PVLDB*, **10**, 1945–1948.

[53] Tschiatschek, S., Singla, A., Gomez Rodriguez, M., Merchant, A., and Krause, A. (2018) Fake news detection in social networks via crowd signals. *WWW '18*, pp. 517–524. WWW.

[54] Pennycook, G. and Rand, D. (2018) *Crowdsourcing judgments of news source quality*. SSRN.

[55] Kim, J., Tabibian, B., Oh, A., Schölkopf, B., and Gomez-Rodriguez, M. (2018) Leveraging the crowd to detect and reduce the spread of fake news and misinformation. *WSDM '18*, pp. 324–332. ACM.

[56] Dong, X. L., Gabrilovich, E., Heitz, G., Horn, W., Murphy, K., Sun, S., and Zhang, W. (2014) From data fusion to knowledge fusion. *Proceedings of the VLDB Endowment*, **7**, 881–892.

[57] Gao, J., Li, Q., Zhao, B., Fan, W., and Han, J. (2015) Truth discovery and crowdsourcing aggregation: A unified perspective. *Proceedings of the VLDB Endowment*, **8**, 2048–2049.

[58] Rekatsinas, T., Joglekar, M., Garcia-Molina, H., Parameswaran, A., and Ré, C. (2017) Slimfast: Guaranteed results for data fusion and source reliability. *Proceedings of the 2017 ACM International Conference on Management of Data*, pp. 1399–1414.

[59] Shiralkar, P., Flammini, A., Menczer, F., and Ciampaglia, G. L. (2017) Finding streams in knowledge graphs to support fact checking. *ICDM '17*, pp. 859–864. IEEE.

[60] Yang, S., Han, F., Wu, Y., and Yan, X. (2016) Fast top-k search in knowledge graphs. *2016 IEEE 32nd international conference on data engineering (ICDE)*, pp. 990–1001. IEEE.

[61] Hamilton, W., Bajaj, P., Zitnik, M., Jurafsky, D., and Leskovec, J. (2018) Embedding logical queries on knowledge graphs. *Advances in neural information processing systems*, pp. 2026–2037.

[62] Ciampaglia, G. L., Shiralkar, P., Rocha, L. M., Bollen, J., Menczer, F., and Flammini, A. (2015) Computational fact checking from knowledge networks. *PloS one*, **10**, e0128193.

[63] Popat, K., Mukherjee, S., Strötgen, J., and Weikum, G. (2017) Where the truth lies: Explaining the credibility of emerging claims on the web and social media. *WWW '17*, pp. 1003–1012.

[64] Jin, Z., Cao, J., Zhang, Y., and Luo, J. (2016) News verification by exploiting conflicting social viewpoints in microblogs. *AAAI '16*, pp. 2972–2978. AAAI Press.

[65] Mukherjee, S. and Weikum, G. (2015) Leveraging joint interactions for credibility analysis in news communities. *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, New York, NY, USA CIKM '15 353–362. Association for Computing Machinery.

[66] Nguyen, N. P., Yan, G., Thai, M. T., and Eidenbenz, S. (2012) Containment of misinformation spread in online social networks. *Proceedings of the 4th Annual ACM Web Science Conference*, New York, NY, USA WebSci '12 213–222. Association for Computing Machinery.

[67] Chen, W., Lakshmanan, L. V. S., and Castillo, C. (2013) *Information and Influence Propagation in Social Networks* Synthesis Lectures on Data Management. Morgan & Claypool Publishers.

[68] Nemhauser, G. L., Wolsey, L. A., and Fisher, M. L. (1978) An analysis of approximations for maximizing submodular set functions—i. *Mathematical Programming*, **14**, 265–294.

[69] Yao, A. C.-C. (1977) Probabilistic computations: Toward a unified measure of complexity. *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, USA SFCS '77 222–227. IEEE Computer Society.

[70] Chen, W., Wang, Y., and Yang, S. (2009) Efficient influence maximization in social networks. *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA KDD '09 199–208. Association for Computing Machinery.