Containment of Misinformation Spread in Online Social Networks

Nam P. Nguyen[1,2], **Guanhua Yan**[2], **My T. Thai**[1], **Stephan Eidenbenz**[2]
[1]Department of Computer and Information Science and Engineering, University of Florida
[2]Information Sciences (CCS-3), Los Alamos National Laboratory
{nanguyen, mythai}@cise.ufl.edu, {nam, ghyan, eidenben}@lanl.gov

ABSTRACT

With their blistering expansions in recent years, popular online social sites such as Twitter, Facebook and Bebo, have become some of the major news sources as well as the most effective channels for viral marketing nowadays. However, alongside these promising features comes the threat of misinformation propagation which can lead to undesirable effects, such as the widespread panic in the general public due to faulty swine flu tweets on Twitter in 2009. Due to the huge magnitude of online social network (OSN) users and the highly clustered structures commonly observed in these kinds of networks, it poses a substantial challenge to efficiently contain viral spread of misinformation in large-scale social networks.

In this paper, we focus on how to limit viral propagation of misinformation in OSNs. Particularly, we study a set of problems, namely the β_T^I-Node Protectors, which aims to find the smallest set of highly influential nodes whose decontamination with good information helps to contain the viral spread of misinformation, initiated from the set I, to a desired ratio $(1-\beta)$ in T time steps. In this family set, we analyze and present solutions including inapproximability result, greedy algorithms that provide better lower bounds on the number of selected nodes, and a community-based heuristic method for the Node Protector problems. To verify our suggested solutions, we conduct experiments on real world traces including *NetHEPT, NetHEPT_WC* and *Facebook* networks. Empirical results indicate that our methods are among the best ones for hinting out those important nodes in comparison with other available methods.

ACM Classification Keywords

H.4 Information Systems Applications: Miscellaneous; D.2.8 Software Engineering: Metrics—*complexity measures, performance measures.*

General Terms

Algorithms, Experimentation.

Author Keywords

Misinformation containment, Online social networks.

WebSci 2012, June 22–24, 2012, Evanston, Illinois, USA.

INTRODUCTION

The huge number of online social networks together with their diversity have drastically changed the landscape of communications and information sharing in the cyber space nowadays. Many people have integrated popular online social sites, such as Facebook and Twitter, into their everyday lives and rely on them as one of their major news sources. For example, the news of the hit on Bin Laden was first broke out on Twitter long before the US president officially announced it on the public media [1], or the recent event *Occupy of Wall Street* has been spread out quickly and widely to a larger population due to its Facebook page [2]. The popularity of OSNs, as a result, is obtained from the convenience as well as efficiency of information dissemination and sharing based on the trust relationships built among their users. Unfortunately, such trust relationships on social networks can possibly be exploited for distributing misinformation or rumors that could potentially cause undesirable effects such as the widespread panic in the general public. For instance, the misinformation of swine flu was observed in Twitter tweets at the outset of the large outbreak in 2009 [3], or the wide spread of the false report that President Obama was killed on the hacked Fox News' Twitter feed in July 2011 [4].

In order for online social networks to serve as a trustworthy channel for disseminating important information, it is crucial to have an effective strategy to contain or limit the viral effect of such misinformation. In particular, we aim to find a tight set of users whose dissemination with "good information" minimizes the devastating effects of misinformation, or in other words, we want to to make sure that most of the network users are aware of the good information by the time the bad one reaches them. Here, the good information is an authorized announcement to correct the corresponding misinformation. In the above examples, good information could be something simple such as "Swine flu rumor is not correct" or "President Obama is still healthy". However, from whom should the good information be disseminated so that the viral effect of misinformation can be contained in a timely manner, especially when the infected sources are known (e.g., hacked Fox News' Twitter feed) or unknown (e.g., tweets about swine flu)?

Due to the huge magnitude of social network users and the highly clustered structures commonly observed in these kinds of networks, it poses a substantial challenge to efficiently contain viral spread of misinformation in large OSNs. Conventional wisdom mainly focuses on immunization which

chooses a set of nodes in the network to immunize in order to disrupt the diffusion process from a graph-theoretic standpoint. In the setting of containing misinformation in OSNs, immunization of certain nodes requires inspecting every message traversing them and stopping those suspected of carrying misinformation. This process itself, however, can be computationally expensive due to the enormous number of messages that spread in a large online social network, e.g., Facebook or Twitter. For example, there were 177 million tweets sent out in a single day on March 11, 2011 [5], and inspecting a tiny URL embedded in a tweet for potential misinformation can be time consuming and inaccurate [6].

Against this backdrop, in this study we consider a scheme that takes a more offensive approach to fight against viral spread of misinformation on social networks. Rather than classifying messages spreading in a network as misinformation or not, this method relies on the similar diffusion mechanism adopted by the misinformation propagation in order to contain it. The key difference, however, is that misinformation often starts from nodes that are less influential and its propagation speed is thus constrained by the trust relationships inherent in the diffusion process from these origins. The containment methods we consider herein, by contrast, aims to find a smallest set of influential people to decontaminate so that the "good" diffusion process starting from them achieves the desirable effect on the spread of misinformation, i.e., the propagation of misinformation is contained in a small fraction $(1-\beta)$ of the whole big network. We call this problem β_T^I-*Node Protector* where β is the desired decontamination threshold, I is the initial infected set (either known or unknown) and T is the time window (either constrained or unconstrained). Here, the superscript I or subscript T is shown only if I is known or T is constrained, respectively

Some attempts on limiting misinformation have been made in earlier works (see the Related Work). The most relevant work to our effort is the one suggested by Budak et al. [7], in which the authors formulated this as an optimization problem, proved its NP-hardness, and then provided approximation guarantees for a greedy solution based on the submodularity property. However, the key differences between our work and theirs are that (1) they impose a k-nodes budget, i.e., the size of the selected set of nodes is constrained by k, and (2) they assume the *high effectiveness propagation*, i.e., the probability for good information spreading is either one or zero, whereas our decontamination model is more general since it allows arbitrary spreading probabilities. Moreover, we provide a far richer framework for studying the problem of containing viral spread on OSNs, where we consider *not only whether the initial set of nodes contaminated by misinformation are known to the defender, but also take into account the time allowance for the defender to contain the misinformation spread*. We thus believe the results from this work offer more insights into how to contain viral spread on OSNs under diverse constraints in practice.

In a nutshell, our main contributions made in this work are summarized as follows. First, we analyze GVS, an algorithm for $\beta-$Node Protector that greedily adds nodes having the best marginal influence to the current solution, and show that this algorithm selects a small fraction of the total nodes extra from the optimal solution by using the famous $(1-1/e)$ approximation factor (Theorem 1). This result, indeed, provides us a better knowledge on the lower bound of the optimal solution in comparison with the $(1+\ln\frac{\beta N}{\epsilon})$ factor suggested in [8]. Second, we show that β_T^I-Node Protector is hard to approximate to a logarithmic factor. In normal graphs, we apply GVS to the network restricted to T-hop neighbors of the initial set I and achieve a slightly better bound for β_T^I-Node Protector problems. Third, we propose an community-based algorithm which returns a good selection of nodes to decontaminate in a timely manner. Finally, we conduct experiments on real-world traces including *NetHEPT*, *NetHEPT_WC* and *Facebook* networks [9]. Empirical results show that both the greedy and community-based algorithms obtain the best results in comparison with other available methods.

RELATED WORK

The information and influence propagation problem on social networks was first studied by Domingos and Richardson in [10]. In this work, they designed viral marketing strategies and analyze the diffusion processes using a data mining approach. Later, Kempe et al. [11] formulated the influence maximization problem on a social network as an optimization problem. In their seminal work, they focused on the linear threshold and independence cascade models and proposed a generalized framework for both of them, as well as proving the problem of influence maximization with a k-node budget admits a $(1-1/e)$ approximation algorithm. Leskovec et al. [12] studied the influence propagation under the detection of outage break-out situation. In particular, they aimed to find the set of nodes in networks to detect the out-break, e.g., the spread of virus, as soon as possible. Chen et al. [13] improve the efficiency of the greedy algorithm and propose new *degree discount* heuristics that is much faster and scalable.

Some attempts have been made in the light of containing the spread of misinformation. For instance, the concept of using benign computer worms to fight against another species has been studied in [14][15]. Most of these works focus on analyzing the performance of active worm containment in the traditional arena of worm propagation, where infectious computers use scanning strategies to find new victims in the IPv4 address space. Dubey et al. [16] conducted a study under the form of a network game focusing on qasi-linear model with various cost and benefit for competing firms. Bharathi et al. [17] modified the independent cascade model to better capture the competing campaigns in the network. Kostka et al. [18], in a game theory point of view, show that the first propagation spreading is not always advantageous. Recently, Budak et al. [7] considered the strategy of using "good" information dissemination campaign to fight against misinformation propagation in social networks. They formulated this as an optimization problem, proved that it is NP-hard, and then provided approximation guarantees for a greedy solution based on the submodularity property. This

problem, indeed, can be considered as one of the Node Protectors, particularly the β_I-Node Protector variant. In [8], Goyal et al. study information dissemination on social network. They provide an greedy algorithm and give a proof of factor $(1 + \ln \frac{\beta N}{\epsilon})$. However, that algorithm is a bicriteria one with an addictive error ϵ on the number of nodes, whereas our analysis suggests a deterministic bound which does not depend on any error parameter.

From the security perspective of OSNs, the information and influence propagation plays an important role in analyzing and designing strategies to counter misinformation such as rumor and computer malware. In [19], for example, Yan et al. conducted a comprehensive study of malware containment strategies, including both user-oriented and network oriented ones, on a moderate-sized online social network. Their work, albeit offering insights into the nature of malware propagation in realistic OSNs, was performed fully from an empirical perspective, and considered only a simple model for malware propagation. Tackling containment of viral spread of misinformation in OSNs, however, demands solutions with a stronger theoretic footing such that they are applicable to a variety of online social network structures and information dissemination models. Moreover, Xu et al. considered the problem of detecting worm propagation in online social networks, and showed that finding a minimal set of nodes to monitor for the purpose of traffic correlation analysis is an NP-complete problem [20]. Our work differs from theirs as we focus on containing, rather than detection, of misinformation propagation in OSNs.

DIFFUSION MODELS AND PROBLEM DEFINITION

In this section, we first define two models of influence propagation in online social networks, as well as a mechanism modeling how good information is disseminated in the network in order to contain misinformation. Under these models, we further formulate the $\beta_T^I - NodeProtector$ problem, which aims to find the smallest set of highly influential nodes in the decontamination campaign.

Propagation Models

We first describe two types of information diffusion in the network, namely the Linear Threshold and Independence Cascade models. These propagation models have received a great attention since the seminal work of Kempe et al. [11] and in this context, they are discussed using the same notations. For the sake of consistency, we call a node *active* if it is influenced by the misinformation either initially or sequentially from one of its neighbors, or *inactive* otherwise.

Linear Threshold (LT) model

In this model, the chance for a node v to adopt the misinformation by a neighbor w is determined according to the weight $b_{v,w}$ that satisfies $\sum_{w \in N(v)} b_{v,w} \leq 1$ for all w in the neighborhood $N(v)$ of v. Initially, each node $v \in V$ independently selects a threshold $\theta_v \in [0, 1]$ uniformly at random. The goal of this threshold is to represent the weighted fraction of v's neighbors that must adopt the misinformation active in order for v to become active. As we shall see next, this threshold is related to a linear constraint of $b_{v,w}$'s, and hence the name of the model. Now, given the chosen thresholds for all nodes v's in V, the propagation progresses from an initial set of infected nodes I in as follows: in step t, all active nodes in step $t-1$ remain active, and any inactive node v for which the total weight of its active neighbors is at least θ_v: $\sum_{w \in N_{active}(v)} b_{v,w} \geq \theta_v$ is activated.

Independence Cascade (IC) model

In this model, any node v that became active in step t will have only one chance to activate each of its currently inactive neighbor w, and the activation from v to a neighbor node u succeeds with a probability $p_{v,w}$. This probability is a parameter of the system and does not depend of the history thus far. If node w has multiple newly activated neighbors, they will try to activate w sequentially in an arbitrary order. If any of these attempts succeeds, w is activated at time step $t+1$ and the same procedure continues further on w, i.e., w will try to activate its inactive neighbors. Again, any node is given only a single chance to influence its friends, thus if it fails to do so in time t, it is not allowed to activate its friends again in time $t+1$.

Decontamination mechanism

The decontamination mechanism in our problem is coincident with the misinformation spreading model. In particular, once the good information spreads out from a particular set of nodes A_I, each node u in A_I will try to spread out the good information to its neighbor node v with the same influence probability $p_{u,v}$ (as in the underlying IC model), or with the same influence threshold θ_v (as in the LT model). We also assume that once a node is decontaminated with good information, it will no longer be influenced by the misinformation. Moreover, once good and misinformation reach a node at the same time, the good information take effect over the bad one. This assumption makes sense for online social networks in reality since the good message could announce to accordingly fix a misinformation included within itself.

Problem Definition

With the discussed independent cascade and linear threshold models as well as the decontamination mechanism taken into account, we consider the following problem in this paper:

DEFINITION 1. *(β_T^I-**Node Protector**) Given an social network represented by a directed graph $G = (V, E)$ and an underlying diffusion model (either LT or IC model). In the presence of misinformation spreading out on G from an either known or unknown initial set I, our goal is to choose the set $S \subseteq V$ of least nodes to decontaminate with good information so that the expected decontamination ratio on the whole network, after T time windows, is at least β. Here $T \in \mathbf{N}$ and $\beta \in [0..1]$ are input parameters.*

Based on the settings of the initial set I and the time window T, we have the following four different variants of Node Protector (NP) problems whose NP-hardness properties can be certified and are omitted here due to space limit.

1. $\beta-$NP: I is unknown and T is unconstrained ($T = \infty$).

2. β^I-NP: I is known and T is unconstrained ($T = \infty$).

3. β^I_T-NP: I is known and T is constrained ($T < \infty$).

4. β_T-NP: I is unknown and T is constrained ($T < \infty$).

Notations

Let $N = |V|$ be the total number of nodes in G. For any node $v \in V$ and any $A \subseteq V$, let $\sigma(v)$ and $\sigma(A)$ be the expected number of nodes that will be influenced by v and A, respectively, if v and A adopt the misinformation (or the dissemination of good information). Note that $\sigma(A)$ and $\sum_{v\in A}\sigma(v)$ are not necessarily be the same in general, and $\sigma(v)$ can contain more nodes than just neighbors of v. In [11], Kempe et al. proved that $\sigma()$ function is submodular under both IC and LT models, and thus the problem of selecting the set S of k nodes that maximizes $\sigma(S)$ admits an $(1 - 1/e)-$approximation guarantee. Other works [7][21] also study this type of problem in some different settings and try to show the submodular property of the $\sigma()$ function in these cases, thus conclude the same guarantee factor.

In a different viewpoint, our problem is the complementary of the previous one where we are given a desired decontamination ratio β and the goal to find the set S of least nodes. We stress that while other studies try to prove the submodular property to get the well-known $(1 - 1/e)$ factor, we indeed use this factor to provide a new point of view into the problem. Specifically, we use this guarantee to derive a better lower-bound on the number of nodes in the optimal solution, as described in the next section.

A BOUNDED METHOD FOR $\beta-$NODE PROTECTOR

When the initial infected set I is unknown and the time window T is unconstrained, β_∞(or simply β)$-$Node Protector asks for the smallest set of nodes whose dissemination of good information helps to achieve at least β percent decontamination ratio *at the end of the process*, where no more nodes get influenced or decontaminated. This is the most general case that usually occurs in practice, especially on large OSNs, and is also the most difficult case to solve. The main source of difficulty here is that the lack of knowledge about the initial set I does not enable us to wisely choose nodes to decontaminate, and thus, we have to do it blindly with the hope that we could have a good solution. Moreover, due to its NP-hardness, it seems unrealistic for one to expect an optimal algorithm for this problem.

A bounded algorithm

We analyze GVS (*Greedy Viral Stopper*), a greedy solution for $\beta-$Node Protector utilizing a modification of the well-known Hill-Climbing (HC) algorithm [22]. At each round of the algorithm, we include a node v adding the maximal marginal gain $\sigma(S + v) - \sigma(S)$ to the current set S until the β fraction of safe nodes is obtained. By doing so, we can show that this solution is within a small amount extra from the optimal solution (Theorem 1). Alg. 1 describes GVS.

Algorithm 1 GVS algorithm for $\beta-$Node Protector

Input: Network $G = (V, E)$, threshold $\beta \in (0, 1]$;
Output: A set $S \subseteq V$ satisfies $\sigma(S) \geq \beta|V|$;
1: $k \leftarrow 1$;
2: $S_k \leftarrow \emptyset$;
3: **while** $(\sigma(S_k) < \beta|V|)$ **do**
4: $\quad v \leftarrow \arg\max_{u\in V\setminus S_k}\{\sigma(S_k + u) - \sigma(S_k)\}$;
5: $\quad k \leftarrow k + 1$;
6: $\quad S_k \leftarrow S_k \cup v$;
7: **end while**
8: Return S.

THEOREM 1. *Alg. 1 returns a solution S of K nodes for $\beta-$Node Protector problem that expectedly satisfies*

$$K \leq |OPT| + \max\big\{0, \frac{\beta N}{\delta e} - \frac{\Delta}{\delta} + 1\big\},$$

where $N = |V|$ is the total number of nodes in the network, $\Delta = \beta N - \sigma(S_{K-1})$, $\delta = \min_{i=1,..,K-2}\{\sigma(S_{i+1}) - \sigma(S_i)\}$ and OPT is an optimal solution set for $\beta-$Node Protector.

PROOF. Let us first describe the notations used in this proof. Let $S_i \quad (i = 1...K)$ be the set produced at the i^{th} step of Alg. 1 (note that $S \equiv S_K$ by this definition). For any integer k, let $Opt_\sigma(k)$ be the maximum value of $\sigma(A)$ over all sets $A \subseteq V$ of $|A| = k$ nodes, i.e., $Opt_\sigma(k) = \max_{A\subseteq V,|A|=k}\{\sigma(A)\}$. Furthermore, let $q = |Q|$ where Q is the optimal solution set with the lowest $\sigma(Q)$ that exceeds βN, i.e., $Q = \arg\min_{A \text{ is an OPT, } \sigma(A)\geq\beta N}\{\sigma(A)\}$. It follows from the above definitions that

(i) $q = |OPT|$ for any optimal solution set OPT.

(ii) $Opt_\sigma()$ and $\sigma()$ are nondecreasing functions, and $\forall A \subseteq V, \sigma(A) \leq Opt_\sigma(|A|)$.

(iii) There lie no $Opt_\sigma(k)$'s ($\forall k = 1...K$) strictly between βN and $\sigma(Q)$ (otherwise it will violate the definition of Q).

We are now ready to prove the theorem. Since Alg. 1 terminates at the K^{th} step, it follows that $\sigma(S_{K-1}) < \beta N$. We consider the following cases:

Case 1: $Opt_\sigma(K-1) < \beta N$. When $Opt_\sigma(K-1) < \beta N$, it implies that any set with even $K-1$ nodes is not sufficient to achieve the desired disinfection goal. Therefore, the optimal solution Q must contain $q > K - 1$ nodes. Moreover, $q \leq |S| = K$ since S is a regular solution. This means $|Q| = K$, which in turns implies that S is also an optimal solution.

Case 2: $\beta N \leq Opt_\sigma(K - 1) \leq \sigma(Q)$. By the definitions of Q, $Opt_\sigma(\cdot)$ and due to (iii), this case can only be valid either when $Opt_\sigma(K - 1) = \beta N$ or $Opt_\sigma(K - 1) = \sigma(Q)$.

When $Opt_\sigma(K - 1) = \beta N$, it follows that the optimal solution Q must span exactly βN nodes since $\sigma(Q)$ is the closest to βN. This also implies $\beta N = Opt_\sigma(K - 1) = Opt_\sigma(K - 2) = \ldots = Opt_\sigma(q) = \sigma(Q)$.

When $Opt_\sigma(K - 1) = \sigma(Q)$, it again infers that $\sigma(Q) = Opt_\sigma(q) = \ldots = Opt_\sigma(K - 1)$ due to (ii). Now, if $q = K - 1$, this greedy algorithm will incur just one more node

than the optimal solution. Otherwise, the analysis of Case 3 can be applied in a very similar manner and consequently, we obtain the same result $K \leq q + (\frac{\beta N}{\delta e} - \frac{\Delta}{\delta} + 1)$ for both situations.

Case 3: $\sigma(Q) < Opt_\sigma(K-1) \leq N$. This is our main case to handle. What we know in this case is $|OPT| = q \leq K-1$, implying $Opt_\sigma(q) \leq Opt_\sigma(K-1)$. We need to bound the size difference between S_{K-1} and S_q, or in other words, bounding $K-1-q$. To do so, we will use the $(1-\frac{1}{e})$ approximation result in [22] (note that all K steps of Alg. 1 follow the HC algorithm, and hence, this guarantee follows naturally), giving $\big(1-\frac{1}{e}\big)Opt_\sigma(q) \leq \sigma(S_q) \leq \sigma(S_{K-1}) = \beta N - \Delta$. Therefore, $\sigma(S_{K-1}) - \sigma(S_q) \leq (\beta N - \Delta) - \big(1-\frac{1}{e}\big)Opt_\sigma(q)$. In addition, since $Opt_\sigma(q) \geq \sigma(Q) \geq \beta N$, the above inequality becomes $\sigma(S_{K-1}) - \sigma(S_q) \leq (\beta N - \Delta) - \big(1-\frac{1}{e}\big)\beta N = \frac{\beta N}{e} - \Delta$.

In order to lower bound $\sigma(S_{K-1}) - \sigma(S_q)$, we observe that every time a node v is added into the solution set S_i, the expected number of decontaminated nodes increases at least by $\min\{\sigma(S_{i+1}) - \sigma(S_i)\} \geq \delta$ for $i = q, .., K-2$. Hence $(K-1-q)\delta \leq \sigma(S_{K-1}) - \sigma(S_q) \leq \frac{\beta N}{e} - \Delta$, which implies $|S| \equiv K \leq |OPT| + (\frac{\beta N}{\delta e} - \frac{\Delta}{\delta} + 1)$. This bound also concludes the proof. □

Remarks

Theorem 1 implies the solution returned by GVS is within a linear factor of βN extra from the optimal solution, given the desired decontamination ratio β. Intuitively, the abitrary selection of any βN nodes in the network is always sufficient for our problem; however, this lower bound implies that GVS, indeed, selects at most $\frac{1}{e} \approx 36\%$ of this many nodes extra from the optimal solution. Moreover, the bigger β, i.e., the smaller the number of misinformation nodes we allowed, the more nodes we have to protect, and vice versa. The bound in Theorem 1 nicely reflects this intuition: when β is bigger, the range for K in the right hand side (RHS) gets larger, which allows K to gets bigger as more nodes needed to be decontaminated. Vice versa, when β gets smaller, the range for K reduces as fewer nodes need to be protected.

ALGORITHMS FOR β^I- AND β_T^I- NODE PROTECTORS

In this section, we study β_T^I-Node Protector problems where the initial infected set I is known and the time step T is either constrained or unconstrained. Due on its NP-hardness, it seems unrealistic for one to find an optimal solution for β_T^I-NP in a timely manner. We further show, by Theorem 2, that this problem in general is hard to approximate to a logarithmic factor via a similar reduction from Set Cover and proof techniques as in [23]. This result tells us how difficult this problem is since it implies the nonexistence of any logarithmic approximation algorithms for β_T^I-Node Protector, under the assumption that $P \neq NP$.

THEOREM 2. *β_T^I-Node Protector can not be approximated in polynomial time to a factor of $c \ln N$, where c is some constant and N is the number of vertices in the graph, under the assumption $P \neq NP$.*

We next present solutions for β^I- and β_T^I-Node Protectors on general networks. The spirit of our approaches for this case is also based on HC algorithm, however, the searching space is significantly reduced due to the knowledge of the initial set I. In particular, we apply GVS algorithm on the network restricted to T-hop neighbors of the initial set I. This provides a slightly better term extra from the optimal solution in comparison with the case of $\beta-$Node Protector. Our approach is based on the following crucial observation: once the infected set is known, the total set of nodes possibly influenced by I is reduced to $N_T(I)$ while the nodes in $V \backslash N_T(I)$ will never be active, where $N_T(I) = \bigcup_{u \in I} N_T(u)$ and

$$N_T(u) = \begin{cases} \{v \in V | u \text{ can reach } v \text{ within } T \text{ hops}\}, & T < \infty \\ \{v \in V | u \text{ can reach } v\}, & T = \infty \end{cases}$$

Hence, once given an initial infected set I and the desired disinfection ratio β in T time windows, the algorithm first identifies $N_T(I)$ and then executes GVS (Alg. 1) on the induced graph $G[N_T(I)]$ with the new disinfection ratio

$$\beta' = \beta - \frac{N - |N_T(I)|}{N},$$

since these $N - |N_T(I)|$ nodes are out of reach of I. If $\beta' \leq 0$, it means that the fraction of nodes outside of $N_T(I)$ is itself sufficient and thus, we do not have to execute the algorithm. Therefore, we focus on the case $\beta' \geq 0$. By using the similar analysis as in the case of $\beta-$Node Protector, we can derive the following result

THEOREM 3. *Alg. 1 on the induced graph $G_{N_T(I)}$ returns a solution S of K nodes for β_T^I-Node Protector that expectedly satisfies*

$$K \leq |OPT| + \max\big\{0, \frac{\beta' |N_T(I)|}{\delta e} - \frac{\Delta'}{\delta} + 1\big\},$$

where $\delta = \min_{i=1,..,K-2}\{\sigma(S_{i+1}) - \sigma(S_i)\}$, $\Delta' = \beta' N - \sigma(S_{K-1})$, and OPT is an optimal solution set for β_T^I-Node Protector problem.

EXPERIMENTAL RESULTS

In this section, we show the experimental results of GVS algorithm on three real networks including the *NetHEPT*, *NetHEPT_WC* and the Facebook social networks. We want to demonstrate the followings (1) how GVS algorithm works on $\beta-$ and β_T^I-Node Protector problems via some practical settings in comparison to other available methods (2) the expected lower bounds of the optimal solutions between ours and those suggested by the $(1 + \ln \frac{\beta N}{\epsilon})$ factor [8].

We also planned to compare our results to those of [7]. However, since the dissemination probabilities in our models are distributed in the range $[0, 1]$ (as of the diffusion model), whereas [7] assumes the high effectiveness property (i.e., good information spreads out with an absolute probability, i.e., $p_{u,v} = 1$ if there is an edge from u to v, and zero otherwise), it does not seem appropriate to do so. In what follows we assume the IC information propagation model.

Datasets

NetHEPT and NetHEPT_WC

The NetHEPT network is a widely-used dataset for testing information diffusion purpose [21][24]. This dataset contains information, mostly the academic collaboration from the "High Energy Physics - Theory" section on arXiv where nodes stand for authors and links represent the coauthorship. In their deliverable, the NetHEPT networks contain 15233 nodes and 31398 links, and the probabilities on edges are assigned by either uniformly at random (for NetHEPT) or by *weighted cascade* (for NetHEPT_WC) where $p(u, v) = 1/d_{in}(v)$ with $d_{in}(v)$ is the indegree of a node v. Note that WC is a special case of IC model when the probability for each edge is predetermined.

Facebook network

This dataset contains friendship information among New Orleans regional network on Facebook, spanning from September 2006 to January 2009 [9]. To collect the information, the authors created several Facebook accounts, joined each to the regional network, started crawling from a single user and visited all friends in a breath-first-search fashion. The data set contains more than 63K nodes (users) connected by more than 1.5 million friendship links with an average node degree of 23.5. In our experiments, the propagation probability for each link connecting to users u and v is proportional to the communication frequency between u and v, normalized on the whole network.

Setup

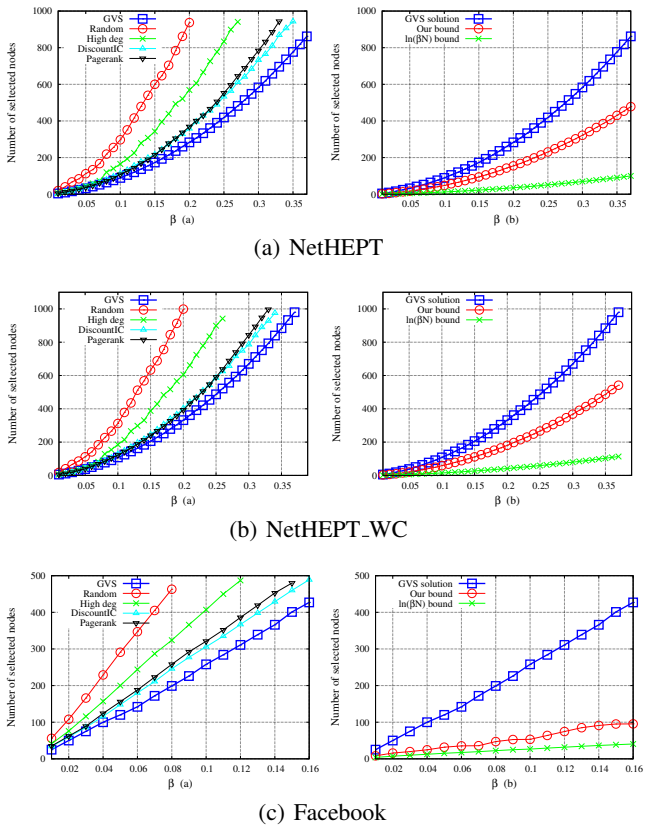
We compare GVS algorithm with the following algorithms (1) *Random*: Include nodes at random to the current solution until the stopping criterion is met (2) *High degree*: Include nodes with the highest weighted degree to the solution until the stopping criterion is met (3) *DiscountIC*: A method based on the weighted discount for IC model suggested in [21] and (4) *Page Rank*: Include nodes to the solution based on their importance.

In all experiments, the Monte Carlo simulation for estimating expected influence is averaged over 1000 runs for consistency. Since the execution of the GVS method is expensive as shown in the running time analysis, we just conduct test cases on small values of β, ranging from 0.01 to 0.37. At any value of β, we run all methods independently and report the number of selected nodes suggested by each of them.

Number of selected nodes

Results on $\beta-$Node Protector

Recall that the ultimate goal of our problem is to choose the set of least nodes so that at least β dissemination ratio on the whole network is achieved, therefore, the smaller number of nodes we have to choose, the better. Left charts of Figures 1(a), (b) and (c) report the performance on $\beta-$Node Protector problem of all methods in different datasets. As depicted in those figures, the number of selected nodes returned by GVS algorithm is the smallest among all competitors. In particular, GVS is roughly 24% better than the second best method DiscountIC, is 75% better than the Highest Degree, and is more than 1.5x better than the Random method.



(a) NetHEPT

(b) NetHEPT_WC

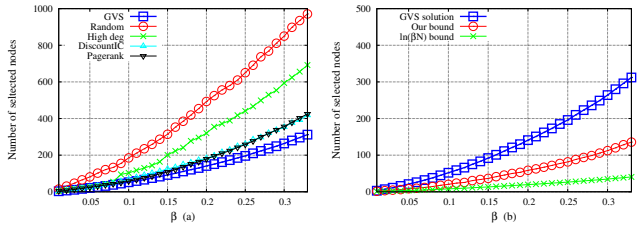
(c) Facebook

Figure 1. Results of GVS algorithm for $\beta-$Node Protector on real social networks (left figures) and the expected lower bounds of the optimal solution (right figures)

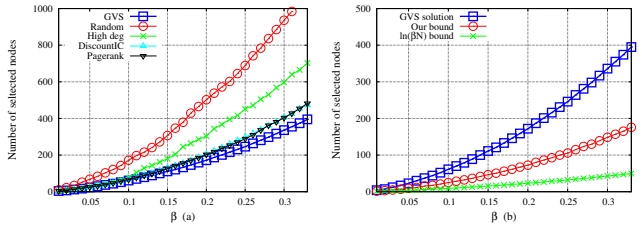
We observe that the behavior of all methods are nearly the same on the two NetHEPT and NetHEPT_WC datasets although the number of selected nodes on the NetHEPT_WC data is slightly smaller. Moreover, the number of selected nodes tends to curve up as the dissemination fraction β gets larger. On Facebook dataset, GVS, again, outperforms other methods in term of size of the selected seeds set and is much better than Random and Highest Degree methods. While Random, not surprisingly, performs the worst in the pool, we had expected Highest Degree to have a better performance. Its poor performance can be explained by the sparseness of the real network: the highest degree nodes could influence more nodes, however, they are not necessarily the most influential ones possibly because they have a little chance to influence each of their neighbors. In fact, this is the case since edges with high probabilities mostly connect low degree nodes in Facebook network. In addition, unlike in other datasets, the numbers of nodes returned by all methods on Facebook network increases linearly as β becomes bigger.

Results on β^I and β^I_T-Node Protectors

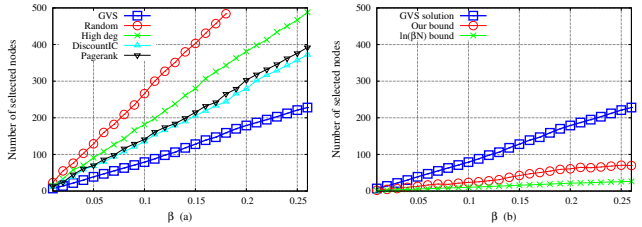
We next look at the behavior of GVS and other methods under β^I and β^I_T-Node Protector settings. In particular, we randomly choose 15% of the total number of nodes to be I, the initial source of misinformation propagation, and set $T = 5$ and $T = \infty$ time windows. We restrict the scope of GVS algorithm on the reduced network $G[N_T(I)]$, and



(a) NetHEPT



(b) NetHEPT_WC



(c) Facebook

Figure 2. Results of GVS algorithm for β_5^I-Node Protector on real social networks (left figures) and the expected lower bounds of the optimal solution (right figures)

provide I as part of the input for other methods. The numbers of nodes - edges contained in these restricted networks $G[N_T(I)]$'s of β_T^I-Node Protector (resp. β^I-) are 6645 - 18236 (resp. 7315 - 22189) for NetHEPT, 8647 - 21091 (resp. 9745 - 26352) for NetHEPT_WC and 21816 - 865930 (resp. 26816 - 1M) for Facebook dataset, respectively.

The results are reported in left charts of Figures 2(a), (b) and (c). As expected, the numbers of selected nodes returned by the GVS algorithm on the reduced networks outperform the other competitors with bigger gaps between them. In average, GVS is 64% better than DiscountIC method, 1.2x better than the Highest Degree method and up to 2.3x better than the worst Random method. The behaviors of all methods are also consistency with what have been observed in the previous test case. We also notice the reduction on numbers of nodes to be selected of all methods, particularly for GVS. The number of nodes chosen by GVS algorithm is reduced by at least one half for three cases. Of course, this is what one should expect once the knowledge of I is provided, and especially when the sizes of the restricted network are much smaller. The empirical result charts for β^I-Node Protector are highly similar to those visualized and discussed above and thus, are excluded for simplicity.

There are rooms for improvement here, such as how to wisely choose nodes when I is the set of some specific targets such as high-degree nodes, or nodes with low or average degrees. While they do sound interesting, we believe that is out of what we are aiming at, and thus, would be included in our future work.

Lower bounds of the optimal solutions

We further investigate on how big the size of the optimal solution set would expectedly be once we are provided with the greedy solution S of GVS. Recall that our result is $|S| \equiv K \leq |OPT| + \max\{0, (\frac{\beta N}{\delta\epsilon} - \frac{\Delta}{\delta} + 1)\}$. Goyal et al. [8] states that $K \leq (1 + \ln \frac{\beta N}{\epsilon})|OPT|$, where $\epsilon \geq 1$ is the additive error to the number of nodes to be decontaminated. Given a solution S of GVS algorithm, both this result and ours provide the knowledge on the lower-bound of the optimal solution OPT. Theoretically, none of them dominates the other on all network instances (for example, a clique with $(1-\eta)$ probabilities on all edges is more suitable for the $(1 + \ln \frac{\beta N}{\epsilon})$-factor while social networks are more suitable for ours, as we shall see below). Of course, a larger and tighter value for this lower bound is of desire since it tells us how large the size of the optimal solution should be. Thus, we also want to know how the two results look like in real-world networks. Here, $\epsilon = 1$ since in term of nodes, this error should be an integer greater or equal to one.

As revealed in the right charts of Figures 1, 2 (a), (b) and (c), our lower bounds (indicated in red circles) are usually larger than that of [8] in all test cases, and consequently provide a more meaningful insight into the expected size of the optimal solution: for any given β, the size of the optimal solution should lie somewhere in between the region induced by the GVS solutions and our lower bounds. This also provides a new point of view into the problem as it ensures that the optimal solution is not too far away from the one returned by the greedy algorithm. Note that our result does not necessarily imply the existence of any constant or logarithmic factor approximation algorithms.

A COMMUNITY-BASED HEURISTIC ALGORITHM

As described in the experiments, GVS algorithm provides very good solutions for both $\beta-$ and β_T^I-Node Protector problems in comparison with other methods. However, one of its down sides is the extremely slow execution due to the expensive task of estimating the marginal influence when a node is added to the current solution. Even with available speed-up provided in [21][24][25] for estimating this marginal gain, GVS still takes a long time to finish its tasks, especially on Facebook social networks (more than 5 hours). This means GVS, despite its very good outcome, might not be the best method for analyzing large-scale online social networks, particularly when the execution time is also a constraint. This drives the need for a more desirable approach which can return a good solution set in a timely manner.

To derive a good heuristic method for Node Protector problems on large OSNs, we take into account a notable phenomenon that commonly exhibits among them: the property of containing community structure, i.e., they naturally divide into groups of vertices with denser connections inside

each group and fewer connections crossing groups, where vertices and connections represent network users and their social interactions, respectively. Roughly speaking, a community on social networks usually consists of people sharing common interests who tend to interact more frequently with other members in the same community than with the outside world. The knowledge of network community structure, as a result, provides us a much better understanding about its topology well as organization principles. Community detection methods and algorithms can be found in an excellent survey of Fortunato et al. [26].

Social-based and community-based algorithms utilizing network communities have been shown to be effective, especially when applied to online social networks [27][28][29]. Taking into account this great advantage of community structure, we propose a community-based heuristic method that can return a reasonable solution in an timely manner. Specifically, our method consists of two main phases (1) *Community detection* phase to quickly reveal the network community structure and (2) *Influence node selection* phase to effectively select nodes of high influence. The detailed algorithm is described in Alg. 2.

Algorithm 2 A community-based algorithm for $\beta-$Node Protector

Input: Network $G = (V, E)$, threshold $\beta \in (0, 1]$;
Output: A set $S \subseteq V$ satisfies $\sigma(S) \geq \beta|V|$;

```
Use Blondel's algorithm [30] to find community structure in G; Let C = {C_1, C_2, ..., C_p} be the network communities with |C_1| ≥ |C_2| ≥ ··· ≥ |C_p|;
S ← ∅;
for i from 1 to p do
    S_i ← ∅;
    while (σ(S_i) < β|C_i|) do
        v ← arg max_{u∈V\S_i} {σ(S_i + u) − σ(S_i)};
        S_i ← S_i ∪ {v};
    end while
    S ← S ∪ S_i;
    if σ(S) ≥ β|V| then
        break;
    end if
end for
Return S.
```

Community detection

As described in Alg. 2, detecting the network communities is the first phase and also is an important part of our method. A precise community structure that naturally reflects the network topology will help the selection of influential nodes in each community to be more effective. Because the detection of network communities is not our focus in this paper, we utilize an community detection method proposed by Blondel et al. [30] whose performance has been verified thoroughly in the literature [31].

There are some good features of this detection algorithm that nicely suite our purposes. First, it returns a community structure whose links within a single community are usually of high probabilities, and those who coming across communities are often of low probabilities. This is to say, the influence propagation within each community is of high probability whereas the chance for misinformation to spread out between communities are relatively small. Second, the size of each community is much smaller in comparison with the whole network, and nodes in each community are usually (weakly) connected to each other. Moreover, they are often of small distances, i.e., they can reach each other within a few hops. Finally, it execution time is reasonably fast, which means it would not add more time to our process as a whole.

High influential node selection

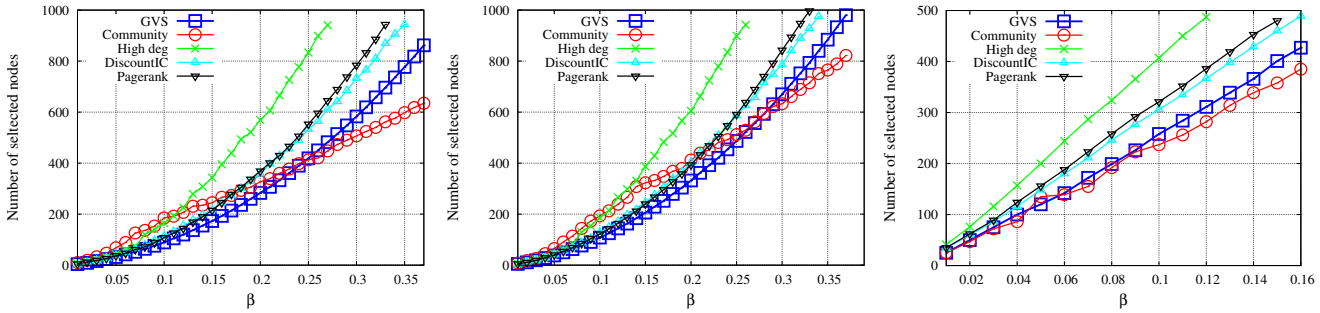
As soon as the first phase finishes, we are provided with a community structure $\mathcal{C}$ as partition of V into disjoint subsets $C_1, C_2, ..., C_p$, and we need to select nodes from these subsets so that the β dissemination ratio is achieved. For simplicity, we assume that the communities are sorted in a non-increasing order of their cardinalities.

Since edges crossing between communities are of usually low probabilities, it follows that a node from a community often has a little chance to spread out misinformation (or good information if it is decontaminated) to another node in a different community. Therefore, our problem can be regarded as the selection of nodes in each community to decontaminate so that β percent of inactive nodes is achieved within each community, and hence achieving the total β percent on the whole network. Intuitively, one would think of applying the GVS algorithm to each community to find the set of influential nodes to decontaminate. In fact, that is the approach we adopt here. At each community C_i, we greedily select nodes providing the maximal marginal gain and add it to S_i as well as the final solution S, until the stopping criterion is met. The motivation behind our approach is due to the stronger of influence within each community and the much smaller size of each community in comparison with the whole network. Thus the selection of influential nodes should not be too expensive as it used to be.

Node selection results

In this subsection, we report the simulation results of the heuristic algorithm in comparison with the aforementioned methods. We demonstrate the followings (1) the number of selected nodes and (2) the execution time. The experimental setup is still kept the same as in Section . The numbers of communities detected by Blondel's algorithm on NetHEPT, NetHEPT_WC and Facebook are 1841, 1839 and 260, respectively. We remove Random method from the charts due to its poor performance and to make the plots more visible.

Simulation results are reported in Figure 3. As depicted in these subfigures, the numbers of nodes selected by the community-based method (Community - in red circles) are highly competitive in comparison with those of other methods, if not to say they tend to get much better as more and more nodes need to be immunized with the misinformation. In particular, this quantity is a little big lag behind the others for small values of $\beta \in [0...0.18]$ on both NetHEPT and NetHEPT_WC datasets and $\beta \in [0...0.9]$ on Facebook network, however, it becomes much better than other methods as the dissemination ratio β gets larger. In average, the community-based method is roughly 16%, 41% and 22% better than GVS, High Degree and DiscountIC methods for $\beta \in [0.2...0.35]$ on NetHEPT and NetHEPT_WC, and is



(a) NetHEPT (b) NetHEPT_WC (c) Facebook

Figure 3. Results of the community based method on social networks

nearly 10% better than the greedy method on Facebook for $\beta \in [0.09...0.16]$. Moreover, this quantity tends to increase linearly in a long run on all of the testing networks, unlike the expensive curvy shapes of the others.

There are reasons behind the up and down sides of the community -based method in our experiments. A scrutiny look into the community structures of the three networks reveals that NetHEPT, NetHEPT_WC and Facebook, in fact, feature only a few large-size communities while containing a lot of small-size ones, most of which are of cardinalities 6 and 8 (Note that this is not a surprising observation since it intuitively agrees with the finding of [26][32]). Moreover, the most influential nodes are usually scattered among different communities. Therefore, when the number of required nodes is small, GVS algorithm can freely pick up these influential nodes while the Community algorithm has to obey selecting influential nodes in larger communities. That explain the down sides of this method in the smaller range of β. However, on the other hand, when there are more and more nodes need to be immunized, Community algorithm can simply pick up most influential nodes from smaller-size communities (where each of which can easily influence the whole community), where GVS algorithm might have to select more nodes from other places to satisfy the criterion.

We next investigate the effect of network communities in our problems. In the general belief, while the local stars are ideal for vaccination, the dissemination should also target on nodes that are accross communities since (1) they are typically a small population, and (2) decontaminating them will keep any misinformation within small circles. However, that is not the case observed in our experiment. The reason can be explained as follow: since the discovered communities are of high internal (and low external) propagation probabilities within each community (and between multiple communities), the bridge nodes are generally of little chances to spread out the misinformation to their neighbor communities, whereas the local stars preserve much higher impact and thus should be included in the solution. This is, perhaps, our most interesting finding that is somewhat in contradiction with the general belief.

Table 1. Running time of five methods

	NetHEPT	NetHEPT_WC	Facebook
GVS	33.1 min	36.1 min	5 hours
Community	**10 sec**	**11 sec**	**2.4 mins**
High Deg	41.8 sec	44.2 sec	4.3 mins
DiscountIC	4.3 min	4.3 min	20.3 mins
PageRank	14.4 min	14.4 min	21 mins

Running time

We next take a look at the running time of the Community algorithm (community detection time is also included) and other methods. In compensation for its good performance, GVS consumes a huge amount of time on each of the datasets, especially on Facebook network where it takes more than 5 hours to find out the solution. While other methods take fair amounts of time (from 1 to 21 minutes) for analyzing these average-sized datasets, they pose potential possibilities to consume much more time on larger social networks. The community-based method, thanks to the advantage of the network community structure, is able to reduce the huge processing time (from hours to minutes) and finishes its tasks in a timely manner, while maintaining a competitive performance. However, unlike GVS, this algorithm does not provide any guarantee on the optimal solution.

In conclusion, we believe that GVS is one of the best informative methods for finding highly influential nodes on small social networks where running time is not a requirement, whereas the community-based method can be regarded as a good heuristic method for hinting out those important nodes on large-scale social networks.

CONCLUSION AND DISCUSSION

We study β_T^I-Node Protector problems which aim to find out the set of least nodes whose decontamination with "good" information provides at least β disinfection ratio on the whole network. We analyze GVS (*Greedy Viral Stopper*), an algorithm for $\beta-$Node Protector that greedily adds nodes with the best influence gain to the current solution, and show that this algorithm selects a small fraction of the total nodes extra from the optimal solution. We apply GVS to the network restricted to T-hop neighbors of the initial set I and achieve a slightly better bound for β_T^I-Node Protector problems.

We propose an community-based algorithm which returns a good selection of nodes to decontaminate in a timely manner. Finally, we verify our approaches on real-world traces including *NetHEPT*, *NetHEPT_WC* and *Facebook* networks.

There are some open issues that are worth discussing here. Firstly, we assume that once a user is disseminated with the good information, he will spread it out to all of his friends. However, in reality, how would we persuade a highly influential node (probably a star) to adopt the good information and spread it out on the social network? Secondly, we assume that the dissemination and misinformation diffusion models are coincident. It would make more sense, in practice, if we use different probabilities for them since the good information may spread out with a faster rate than the misinformation. Nevertheless, we find these issues interesting and would be included in our research directions in the future.

Acknowledgment

This work is partially supported by the DTRA YIP grant number HDTRA1-09-1-0061, and the Los Alamos National Laboratory Directed Research and Development Program, project number 20110093DR.

REFERENCES

1. http://articles.cnn.com/2011-05-02/tech/osama.bin.laden.twitter_1_bin-tweet-twitter-user?_s=PM:TECH.
2. http://www.facebook.com/OccupyWallSt.
3. www.pcworld.com/businesscenter/article/163920/swine_flu_frenzy_demonstrates_twitters_achilles_heel.html.
4. http://articles.cnn.com/2011-07-04/tech/fox.hack_1_tweets-twitter-feed-twitter-users?_s=PM:TECH.
5. http://blog.twitter.com/2011/03/numbers.html.
6. C. Grier, K. Thomas, V. Paxson, and M. Zhang. @spam: the underground on 140 characters or less. In *CCS*, 2010.
7. C. Budak, D. Agrawal, and A. El Abbadi. Limiting the spread of misinformation in social networks. In *WWW*, 2011.
8. A. Goyal, F. Bonchi, and L. V. S. Lakshmanan. Approximation analysis of influence spread in social networks. *arXiv:1008.205.v3*, 2010.
9. B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi. On the evolution of user interaction in facebook. In *WOSN*, 2009.
10. P. Domingos and M. Richardson. Mining the network value of customers. In *KDD*, 2001.
11. D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. *KDD*, 2003.
12. J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective outbreak detection in networks. In *KDD*, 2007, 2007.
13. N. Chen. On the approximability of influence in social networks. *SIAM Journal of Discrete Mathematics*, 23(3), 2009.
14. D. Nicol and M. Liljenstam. Models and analysis of active worm defense. *MMM-ACNS*, 2005.
15. Sapon Tanachaiwiwat and Ahmed Helmy. Encounter-based worms: Analysis and defense. *Ad Hoc Network*, 2009.
16. P. Dubey, R. Garg, and B. D. Meyer. Competing for customers in a social network: The quasi-linear case. *WINE*, 2006.
17. S. Bharathi, D. Kempe, and M. Salek. Competitive influence maximization in social networks. *WINE*, 2007.
18. J. Kostka, Y. A. Oswald, and R. Wattenhofer. Word of mouth: Rumor dissemination in social networks. *SIROCCO*, 2008.
19. G. Yan, G. Chen, S. Eidenbenz, and N. Li. Malware propagation in online social networks: nature, dynamics, and defense implications. In *ASIACCS*, 2011.
20. W. Xu, F. Zhang, and S. Zhu. Toward worm detection in online social networks. In *ACSAC*, 2010.
21. W. Chen, C. Wang, and Y. Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *KDD*, 2010.
22. G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of the approximatios for maximizing submodular set functions. *Mathematical Programming*, 1978.
23. T. N. Dinh, D. T. Nguyen, and M. T. Thai. Cheap, easy, and massively effective viral marketing in social networks: Truth or fiction? *HYPERTEXT*, 2012.
24. W. Chen, Y. Yuan, and L. Zhang. Scalable influence maximization in social networks under the linear threshold model. In *ICDM*, 2010.
25. M. Kimura, K. Saito, R. Nakano, and H. Motoda. Extracting influential nodes on a social network for information diffusion. *Data Min. Knowl. Disc*, 2010.
26. S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5):75 – 174, 2010.
27. T. N. Dinh, Y. Xuan, and M. T. Thai. Towards social-aware routing in dynamic communication networks. *IPCCC*, 2009.
28. N. P. Nguyen, T. N. Dinh, Y. Xuan, and M. T. Thai. Adaptive algorithms for detecting community structure in dynamic social networks. *INFOCOM*, 2011.
29. N. P. Nguyen, T. N. Dinh, S. Tokala, and My T. Thai. Overlapping communities in dynamic networks: their detection and mobile applications. In *MOBICOM*, 2011.
30. V. D. Blondel, J. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *J. Stat. Mech.: Theory and Experiment*, 2008.
31. A. Lancichinetti and S. Fortunato. Community detection algorithms: A comparative analysis. *Physical review. E. 80*, 2009.
32. M. A. Porter, J-P. Onnela, and P. J. Mucha. Communities in networks. *Notices of the AMS*, 56(9), 2009.