Node Immunization on Large Graphs: Theory and Algorithms

Chen Chen*, Hanghang Tong*, B. Aditya Prakash†, Charalampos Tsourakakis¶,
Tina Eliassi-Rad§, Christos Faloutsos‡ and Duen Horng Chau♯

Abstract—Given a large graph, like a computer communication network, which k nodes should we immunize (or monitor, or remove), to make it as robust as possible against a computer virus attack? This problem, referred to as the Node Immunization problem, is the core building block in many high-impact applications, ranging from public health, cybersecurity to viral marketing. A central component in Node Immunization is to find the best k bridges of a give graph. In this setting, we typically want to determine the relative importance of a node (or a set of nodes) within the graph, for example, how valuable (as a bridge) a person or a group of persons is in a social network.

First of all, we propose a novel 'bridging' score $\Delta\lambda$, inspired by immunology, and we show that its results agree with intuition for several realistic settings. Since the straightforward way to compute $\Delta\lambda$ is computationally intractable, we then focus on the computational issues and propose a surprisingly efficient way ($O(nk^2 + m)$) to estimate it. Experimental results on real graphs show that (1) the proposed 'bridging' score gives mining results consistent with intuition; and (2) the proposed fast solution is up to *7 orders of magnitude* faster than straightforward alternatives.

Index Terms—Immunization, Graph Mining, Scalability

1 INTRODUCTION

GIVEN a graph, we want to quickly find the k best nodes to immunize (or, equivalently, remove), to make the remaining nodes to be most robust to the virus attack. This is the core problem for many applications: In a computer network intrusion setting, we want the k best nodes to defend (e.g., through expensive and extensive vigilance), to minimize the spread of malware. Similarly, in a law-enforcement setting, given a network of criminals, we want to neutralize those nodes that will maximally scatter the graph.

There are three main challenges behind this problem. First (*C1. Vulnerability measure*), we need a '*Vulnerability*' measure of the graph, that is, how likely/easily that a graph will be infected by a virus. Second (*C2. Shield-value*), based on the '*Vulnerability*' measure of the entire graph, we further need a measure to quantify the '*Shield-value*' of a given *set* of nodes in the graph, i.e., how important are they in terms of maintaining the '*Vulnerability*' of the graph? Alternatively, how much less vulnerable will be the graph to the virus attack, if those nodes are removed/immunized?

Third (*C3. Algorithms*), based on the '*Shield-value*' measure of a set of nodes, we need an effective and scalable algorithm to quickly determine the k nodes that *collectively* exhibit the highest '*Shield-value*' score on large, disk-resident graphs.

In this paper, we aim to address these challenges in multiple dimensions. Motivated from immunology and graph loop/path capacity, we adopt the first[1] eigenvalue λ of the graph as the '*Vulnerability*' measurement (for C1). Based on that, we propose a novel definition of the '*Shield-value*' score $\mathrm{Sv}(\mathcal{S})$ for a specific set of nodes (for C2). By carefully using the results from the theory of matrix perturbation, we show that the proposed '*Shield-value*' gives a good approximation of the corresponding eigen-drop (i.e., the decrease of the '*Vulnerability*' measurement if we remove/immunize the set of nodes $\mathcal{S}$ from the graph). Furthermore, we show that the proposed '*Shield-value*' score is *sub-modular*, which enables us to develop a *near-optimal* and *scalable* algorithm (*NetShield*) to find a set of nodes with highest '*Shield-value*' score (for C3). Finally, we propose a variant (*NetShield+*) to further balance the optimization quality and computational cost.

The main contributions of this paper can be summarized as

1. A novel definition of the '*Shield-value*' score $\mathrm{Sv}(\mathcal{S})$ for a set of nodes, by carefully using the results from the theory of matrix perturbation.
2. A *near-optimal* and *scalable* algorithm (*NetShield*) and its variant (*NetShield+*) to find a set of nodes with highest '*Shield-value*' score, by carefully using results

- * *are with Arizona State University, Tempe, AZ, USA.*
  E-mail: cchen211@asu.edu, hanghang.tong@asu.edu
- † *is with Virginia Tech, Blacksburg, VA, USA.*
  E-mail: badityap@cs.vt.edu
- ¶ *is with Harvard University, Cambridge, MA, USA.*
  E-mail: babis@seas.harvard.edu
- § *is with Rutgers University, Piscataway, NJ, USA.*
  E-mail: tina@eliassi.org
- ‡ *is with Carnegie Mellon University, Pittsburgh, PA, USA.*
  E-mail: christos@cs.cmu.edu
- ♯ *is with Georgia Tech, Atlanta, GA, USA.*
  E-mail: polo@gatech.edu

1. In this paper, the first eigenvalue means the eigenvalue with the largest module.

from the theory of sub-modularity.

3. Extensive experiments on several real data sets, illustrating the effectiveness and efficiency of the proposed methods.

The rest of the paper is organized as follows: Section 2 gives the problem definitions. We present the 'Vulnerability' measurement in Section 3. The proposed 'Shield-value' score is presented in Section 4. We address the computational issues in Section 5 and evaluate the proposed methods in Section 6. Section 7 gives the related work, and Section 8 gives the conclusions.

2 PROBLEM DEFINITIONS

Table 1 lists the main symbols we use throughout the paper. In this paper, we focus on un-directed un-weighted graphs. We represent the graph by its adjacency matrix. Following standard notations, we use capital bold letters for matrices (e.g., $\mathbf{A}$), lower-case bold letters for vectors (e.g., $\mathbf{a}$), and calligraphic fonts for sets (e.g., $\mathcal{S}$). We denote the transpose with a prime (i.e., $\mathbf{A}'$ is the transpose of $\mathbf{A}$), and we use parenthesized superscripts to denote the corresponding variable after deleting the nodes indexed by the superscripts. For example, λ is the first eigenvalue of $\mathbf{A}$, then λ^i is the first eigenvalue of $\mathbf{A}$ after deleting its $i^{\text{(th)}}$ row/column. We use $(\lambda_i, \mathbf{u}_i)$ to denote the ith eigen-pair (sorted by the magnitude of the eigenvalue) of $\mathbf{A}$. When the subscript is omitted, we refer to them as the first eigenvalue and eigenvector respectively (i.e., $\lambda \triangleq \lambda_1$ and $\mathbf{u} \triangleq \mathbf{u}_1$).

TABLE 1: Symbols

Symbol	Definition and Description
$\mathbf{A}, \mathbf{B}, \dots$	matrices (bold upper case)
$\mathbf{A}(i,j)$	the element at the ith row and j^{th} column of matrix $\mathbf{A}$
$\mathbf{A}(i,:)$	the ith row of matrix $\mathbf{A}$
$\mathbf{A}(:,j)$	the jth column of matrix $\mathbf{A}$
$\mathbf{A}'$	transpose of matrix $\mathbf{A}$
$\mathbf{a}, \mathbf{b}, \dots$	column vectors
$\mathcal{S}, \mathcal{T}, \dots$	sets (calligraphic)
n	number of nodes in the graph
m	number of edges in the graph
$(\lambda_i, \mathbf{u}_i)$	the ith eigen-pair of $\mathbf{A}$
λ	first eigenvalue of $\mathbf{A}$ (i.e., $\lambda \triangleq \lambda_1$)
$\mathbf{u}$	first eigenvector of $\mathbf{A}$ (i.e., $\mathbf{u} \triangleq \mathbf{u}_1$)
$\lambda^{(i)}, \lambda^{(\mathcal{S})}$	first eigenvalue of $\mathbf{A}$ by deleting node i (or the set of nodes in $\mathcal{S}$)
$\Delta\lambda(i)$	eigen-drop: $\Delta\lambda(i) = \lambda - \lambda^{(i)}$
$\Delta\lambda(\mathcal{S})$	eigen-drop: $\Delta\lambda(\mathcal{S}) = \lambda - \lambda^{(\mathcal{S})}$
$\text{Sv}(i)$	'Shield-value' score of node i
$\text{Sv}(\mathcal{S})$	'Shield-value' score of nodes in $\mathcal{S}$
$\text{V}(\mathbf{G})$	'Vulnerability' score of the graph

With the above notations, our problems can be formally defined as follows:

Problem 1: Measuring 'Vulnerability'

Given: A large un-directed un-weighted connected graph G with adjacency matrix $\mathbf{A}$;

Find: A single number $\text{V}(\mathbf{G})$, reflecting the 'Vulnerability' of the whole graph.

Problem 2: Measuring 'Shield-value'

Given: A subset $\mathcal{S}$ with k nodes in a large un-directed un-weighted connected graph G with adjacency matrix $\mathbf{A}$;

Find: A single number $\text{Sv}(\mathcal{S})$, reflecting the 'Shield-value' of these k nodes (that is, the benefit of their removal/immunization to the vulnerability of the graph).

Problem 3: Finding k Nodes of Best 'Shield-value'

Given: A large un-directed un-weighted connected graph G with n nodes and an integer k;

Find: A subset $\mathcal{S}$ of k nodes with the highest 'Shield-value' score among all $\binom{n}{k}$ possible subsets.

In the next three sections, we present the corresponding solutions respectively.

3 BACKGROUND: OUR SOLUTION FOR PROBLEM 1

As mentioned in Section 1, the ultimate goal of Node Immunization problem is to contain epidemic over the network. In an epidemic network, nodes can have different states depending on the epidemic model. The model we simulate here is SIS model [46]. In SIS model, each node would have one of the following two states: susceptible and infected. Susceptible nodes can be infected by infected nodes with infection rate b at each time stamp, and each infected node can get back to susceptible state with host-recovery rate d. Epidemic threshold is an intrinsic property of a network. When the strength of the virus is greater than the epidemic threshold, then the epidemic would breakout.

Here, we begin to address Problem 1. According to [46], the epidemic thresholds of arbitrary cascade models on arbitrary networks can be determined by the largest eigenvalue of network's connectivity matrix. The intuition is that, the larger the largest eigenvalue is, the more connected the graph is, and therefore the more vulnerable the structure is under epidemic. Thus we suggest using the first eigenvalue λ as 'Vulnerability' score. We should point out that it is *not* our main contribution to adopt λ as the 'Vulnerability' measure of a graph. Nonetheless, it is the base of our proposed solutions for both Problem 2 and Problem 3.

3.1 'Vulnerability' Score

In Problem 1, the goal is to measure the 'Vulnerability' of the whole graph by a single number. We adopt the first eigenvalue of the adjacency matrix $\mathbf{A}$ as such a measurement (eq. (1)): the larger λ is, the more vulnerable the whole graph is.

$$\text{V}(\mathbf{G}) \triangleq \lambda \tag{1}$$

Figure 1 presents an example, where we have four graphs with 5 nodes. Intuitively, the graph becomes more and more vulnerable from the left to the right. In other words, for a given strength of the virus attack, it is more likely that an epidemic will break out in the graphs on the right than those on the left side. Therefore, the vulnerability of the graph increases. We can see that the corresponding λ increases from left to right as well. Note that 'Vulnerability' score
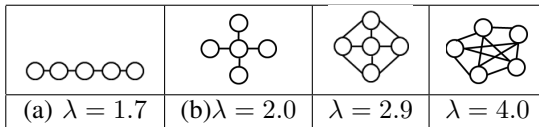
Fig. 1: *An example of measuring 'Vulnerability' of the graph. More edges, and carefully placed, make the graph better connected, and thus more vulnerable. Notice that the chain (a) and the star (b) have the same number of edges, but our λ score correctly considers the star as more vulnerable.*

in this paper is not necessarily comparable between graphs with different number of nodes. That means if we have two graphs with the same '*Vulnerability*' score but different number of nodes, this does not necessarily means that they two have the same ability to contain the epidemic.

Notice that the concept of '*Vulnerability*' is *different* from vertex connectivity of the graph [20]. For '*Vulnerability*', we want to quantify how likely/easiy a graph will be infected by a virus (given the strength of virus attack). Whereas for vertex connectivity, we want to quantify how difficult for a graph to be disconnected. For example, both graph (a) and (b) in figure 1 have the same vertex connectivity (both are 1). But graph (b) is more vulnerable to the virus attack. Also notice that although '*Vulnerability*' is related to both graph density (i.e., average degree) and diameter, neither of them can *fully* describe the '*Vulnerability*' by itself. For example, in figure 1, (a) and (b) share the same density/average degree although (b) is more vulnerable than (a); (b) and (c) share the same diameter although (c) is more vulnerable than (b).

3.2 Justifications

The first eigenvalue λ is a good measurement of the graph '*Vulnerability*', because of recent results on *epidemic thresholds* from immunology [7]: λ is closely related to the epidemic threshold τ of a graph under a flu-like SIS (susceptible-infective-susceptible) epidemic model, and specifically $\tau = 1/\lambda$. This means that a virus less infective than τ will quickly get extinguished instead of lingering forever. Therefore, given the strength of the virus (that is, the infection rate and the host-recovery rate), an epidemic is more likely for a graph with larger λ.

We can also show that the first eigenvalue λ is closely related to the so-called *loop capacity* and the *path capacity* of the graph, that is, the number of loops and paths of length l ($l = 2, 3, \ldots$). If a graph has many such loops and paths, then it is well connected, and thus more vulnerable (i.e., it is easier for a virus to propagate across the graph = the graph is less robust to virus attack). Note that although there are many other measurements that are also related to graph connectivity like second smallest eigenvalue of the Laplacian Matrix of the graph, they are not as directly related to epidemic threshold as λ is, as shown in [46]. Thus, for the epidemic-like influence process, λ is more suitable for evaluating vulnerability score than those alternative measurements.

4 OUR SOLUTION FOR PROBLEM 2

In this section, we focus on Problem 2. We first present our solution, and then provide justifications.
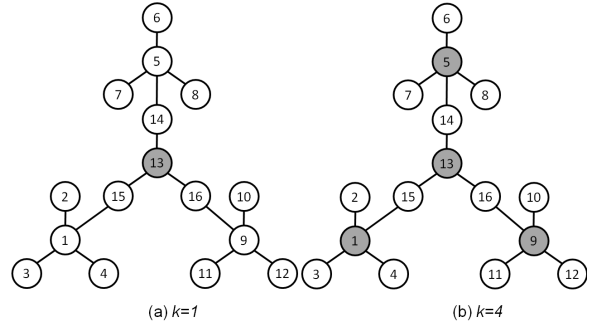
4.1 Proposed '*Shield-value*' Score

Fig. 2: *An example on measuring the 'Shield-value' score of a given set of nodes. The best k nodes found by our NetShield are shaded. In (a), notice that the highest degree nodes (e.g., node 1) is not chosen. In (b), immunizing the shaded nodes makes the remaining graph most robust to the virus attack.*

In Problem 2, the goal is to quantify the importance of a given set of nodes $\mathcal{S}$, and specifically the impact of their deletion/immunization to the '*Vulnerability*' of the rest of the graph. The obvious choice is the drop in eigenvalue, or *eigen-drop* $\Delta\lambda$ that their removal will cause to the graph. We propose to approximate it, to obtain efficient computations, as we will describe later. Specifically, we propose using $\mathrm{Sv}(\mathcal{S})$ defined as:

$$\mathrm{Sv}(\mathcal{S}) = \sum_{i \in \mathcal{S}} 2\lambda \mathbf{u}(i)^2 - \sum_{i,j \in \mathcal{S}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j) \qquad (2)$$

Intuitively, by eq. (2), a set of nodes $\mathcal{S}$ has higher '*Shield-value*' score if (1) each of them has a high eigen-score ($\mathbf{u}(i)$), and (2) they are dissimilar with each other (small or zero $\mathbf{A}(i,j)$). Figure 2 shows an example on measuring the '*Shield-value*' score of a given set of nodes. The best k nodes found by our *NetShield* (which will be introduced very soon in the next section) are shaded. The result is consistent with intuition. In figure 2(a), it picks node 13 as best $k = 1$ node (although nodes 1, 5 and 9 have the highest degree). In figure 2(b), deleting the shaded nodes (node 1, 5, 9 and 13) will make the graph least vulnerable (i.e., the remaining graphs are sets of isolated nodes; and therefore it is most robust to virus attack).

4.2 Justifications

Here, we provide some justifications on the proposed '*Shield-value*' score, which is summarized in Lemma 1. It says that our proposed '*Shield-value*' score $\mathrm{Sv}(\mathcal{S})$ is a good approximation for the eigen-drop $\Delta\lambda(\mathcal{S})$ when deleting the set of nodes $\mathcal{S}$ from the original graph $\mathbf{A}$.

Lemma 1: Let $\lambda^{(\mathcal{S})}$ be the (exact) first eigen-value of $\hat{\mathbf{A}}$, where $\hat{\mathbf{A}}$ is the perturbed version of $\mathbf{A}$ by removing all of

its rows/columns indexed by set $\mathcal{S}$. Let $\delta = \lambda - \lambda_2$ be the eigen-gap, and d be the maximum degree of $\mathbf{A}$. If λ is the simple first eigen-value of $\mathbf{A}$, and $\delta \geq 2\sqrt{2kd}$, then

$$\Delta\lambda(\mathcal{S}) = \mathrm{Sv}(\mathcal{S}) + O(\sum_{j \in \mathcal{S}} \|\mathbf{A}(:,j)\|^2) \qquad (3)$$

where $\mathrm{Sv}(\mathcal{S})$ is computed by eq. (2) and $\Delta\lambda(\mathcal{S}) = \lambda - \lambda^{(\mathcal{S})}$. **Proof**: First, let us write $\hat{\mathbf{A}}$ as a perturbed version of the original matrix $\mathbf{A}$:

$$\hat{\mathbf{A}} = \mathbf{A} + \mathbf{E}, \quad \text{and} \quad \mathbf{E} = \mathbf{F} + \mathbf{F}' + \mathbf{G} \qquad (4)$$

where $\mathbf{F}(:,j) = -\mathbf{A}(:,j)$ $(j \in \mathcal{S}$ and $\mathbf{F}(:,j) = 0$ $(j \notin \mathcal{S})$; $\mathbf{G}(i,j) = \mathbf{A}(i,j)$ $(i,j \in \mathcal{S})$ and $\mathbf{G}(i,j) = 0 (i \notin \mathcal{S}$, or $j \notin \mathcal{S})$.

Since $\mathbf{A}\mathbf{u} = \lambda\mathbf{u}$, we have

$$\begin{aligned}
\mathbf{u}'\mathbf{F}'\mathbf{u} &= \mathbf{u}'\mathbf{F}\mathbf{u} = (\mathbf{F}'\mathbf{u})'\mathbf{u} = -\sum_{j \in \mathcal{S}} \lambda\mathbf{u}(j)^2 \\
\mathbf{u}'\mathbf{G}\mathbf{u} &= \sum_{i,j \in \mathcal{S}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j) \qquad (5)
\end{aligned}$$

Let $\tilde{\lambda}$ be the corresponding perturbed eigen-value of λ, according to the matrix perturbation theory (p.183 [53]), we have

$$\begin{aligned}
\tilde{\lambda} &= \lambda + \mathbf{u}'\mathbf{E}\mathbf{u} + O(\|\mathbf{E}\|_F^2) \\
&= \lambda + \mathbf{u}'\mathbf{F}\mathbf{u} + \mathbf{u}'\mathbf{F}'\mathbf{u} + \mathbf{u}'\mathbf{G}\mathbf{u} + O(\|\mathbf{E}\|_F^2) \\
&= \lambda - (\sum_{j \in \mathcal{S}} 2\lambda\mathbf{u}(j) - \sum_{i,j \in \mathcal{S}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j)) \\
&\quad + O(\sum_{j \in \mathcal{S}} \|\mathbf{A}(:,j)\|^2) \\
&= \lambda - \mathrm{Sv}(\mathcal{S}) + O(\sum_{j \in \mathcal{S}} \|\mathbf{A}(:,j)\|^2) \qquad (6)
\end{aligned}$$

Let $\tilde{\lambda}_i (i = 2, ..., n)$ be the corresponding perturbed eigen-value of $\lambda_i (i = 2, ..., n)$. Again, by the matrix perturbation theory (p.203 [53]), we have

$$\begin{aligned}
\tilde{\lambda} &\geq \lambda - \|\mathbf{E}\|_2 \geq \lambda - \|\mathbf{E}\|_F \geq \lambda - \sqrt{2kd} \\
\tilde{\lambda}_i &\leq \lambda_i + \|\mathbf{E}\|_2 \leq \lambda_i + \|\mathbf{E}\|_F \leq \lambda_i + \sqrt{2kd} \quad (7)
\end{aligned}$$

Since $\delta = \lambda - \lambda_2 \geq 2\sqrt{2kd}$, we have $\tilde{\lambda} \geq \tilde{\lambda}_i (i = 2, ..., n)$. In other words, we have $\lambda^{(\mathcal{S})} = \tilde{\lambda}$. Therefore,

$$\begin{aligned}
\Delta\lambda(\mathcal{S}) &= \lambda - \lambda^{(\mathcal{S})} = \lambda - \tilde{\lambda} \\
&= \mathrm{Sv}(\mathcal{S}) + O(\sum_{j \in \mathcal{S}} \|\mathbf{A}(:,j)\|^2) \qquad (8)
\end{aligned}$$

which completes the proof.□

Notice that $\|\mathbf{E}\|_F$ and $\|\mathbf{E}\|_2$ refer to the Frobenious norm and the l_2 norm of $\mathbf{E}$, respectively. The former is defined as $\|\mathbf{E}\|_F = \sum_{i=1}^n \sum_{j=1}^n a_{ij}^2$, while $\|\mathbf{E}\|_2$ equals to the largest eigenvalue of $\mathbf{E}$. And the inequality $\|\mathbf{E}\|_F > \|\mathbf{E}\|_2$ always holds.

5 OUR SOLUTION FOR PROBLEM 3

In this section, we deal with Problem 3. Here, the goal is to find a subset of k nodes with the highest '*Shield-value*' score (among all $\binom{n}{k}$ possible subsets). We start by showing that the two straightforward methods (referred to as 'Com-Eigs', and 'Com-Eval') are computationally intractable. Then, we present the proposed *NetShield* algorithm and analyze its accuracy as well as its computational complexity. Finally to further balance the optimization quality and computational cost, we propose a variant of *NetShield*, *NetShield+*.

5.1 Challenges

There are two obviously straightforward methods for Problem 3. The first one (referred to as 'Com-Eigs'[2]) works as follows: for each possible subset $\mathcal{S}$, we delete the corresponding rows/columns from the adjacency matrix $\mathbf{A}$; compute the first eigenvalue of the new perturbed adjacency matrix; and finally output the subset of nodes which has the smallest eigenvalue (therefore has the largest eigen-drop). Despite the simplicity of this strategy, it is computational intractable due to its combinatorial nature. It is easy to show that the computational complexity of 'Com-Eigs' is $O((\binom{n}{k} \cdot m)^3)$. This is computationally intractable even for small graphs. For example, in a graph with 1K nodes and 10K edges, suppose that it takes about 0.01 second to find its first eigenvalue. Then we need about 2,615 years to find the best-5 nodes with the highest '*Shield-value*' score!

A more reasonable (in terms of speed) way to find the best-k nodes is to evaluate $\mathrm{Sv}(\mathcal{S})$, rather than to compute the first eigenvalue $\lambda^{(\mathcal{S})}$, $\binom{n}{k}$ times, and pick the subset with the highest $\mathrm{Sv}(\mathcal{S})$. We refer to this strategy as 'Com-Eval'. Compared with the straightforward method (referred to as 'Com-Eigs', which is $O(\binom{n}{k} \cdot m)$); 'Com-Eval' is much faster $(O(\binom{n}{k} \cdot k^2))$. However, 'Com-Eval' is still not applicable to real applications due to its combinatorial nature. Again, in a graph with 1K nodes and 10K edges, suppose that it only takes about 0.00001 second to evaluate $\mathrm{Sv}(\mathcal{S})$ once. Then we still need about 3 months to find the best-5 nodes with the highest '*Shield-value*' score!

Theorem 1: K-Node Immunization with λ is NP complete.
Proof: See the appendix. □

5.2 Proposed *NetShield* Algorithm

The proposed *NetShield* is given in Alg. 1. In Alg. 1, we compute the first eigenvalue λ and the corresponding eigenvector $\mathbf{u}$ in step 1. In step 4, the $n \times 1$ vector $\mathbf{v}$ measures the '*Shield-value*' score of each individual node. Then, in each iteration of steps 6-17, we greedily select one more node and add it into set $\mathcal{S}$ according to score(j) (step 13). Note that steps 10-12 are to exclude those nodes that are already in the selected set $\mathcal{S}$.

2. To our best knowledge, this is the best known method to get the optimal solution of Problem 3.

3. We assume that k is relatively small compared with n and m (e.g., tens or hundreds). Therefore, after deleting k rows/columns from $\mathbf{A}$, we still have $O(m)$ edges.

Algorithm 1 *NetShield*

Input: the adjacency matrix $\mathbf{A}$ and an integer k
Output: a set $\mathcal{S}$ with k nodes

1: compute the first eigenvalue λ of $\mathbf{A}$; let $\mathbf{u}$ be the corresponding eigenvector $\mathbf{u}(j)(j = 1, ..., n)$;
2: initialize $\mathcal{S}$ to be empty;
3: **for** $j = 1$ to n **do**
4: $\quad \mathbf{v}(j) = (2 \cdot \lambda - \mathbf{A}(j, j)) \cdot u(j)^2$;
5: **end for**
6: **for** $iter = 1$ to k **do**
7: $\quad$ let $\mathbf{B} = \mathbf{A}(:, \mathcal{S})$;
8: $\quad$ let $\mathbf{b} = \mathbf{B} \cdot \mathbf{u}(\mathcal{S})$;
9: $\quad$ **for** $j = 1$ to n **do**
10: $\quad\quad$ **if** $j \in \mathcal{S}$ **then**
11: $\quad\quad\quad$ let $score(j) = -1$;
12: $\quad\quad$ **else**
13: $\quad\quad\quad$ let $score(j) = \mathbf{v}(j) - 2 \cdot \mathbf{b}(j) \cdot \mathbf{u}(j)$;
14: $\quad\quad$ **end if**
15: $\quad$ **end for**
16: $\quad$ let $i = argmax_j score(j)$, add i to set $\mathcal{S}$;
17: **end for**
18: return $\mathcal{S}$.

5.3 Analysis of *NetShield*

Here, we analyze the accuracy and efficiency of the proposed *NetShield*.

First, according to the following theorem, Alg. 1 is *near-optimal* wrt 'Com-Eval'. In addition, by Lemma 1, our '*Shield-value*' score (which 'Com-Eval' tries to optimize) is a good approximation for the actual eigen-drop $\Delta\lambda(\mathcal{S})$ (which 'Com-Eigs' tries to optimize). Therefore, we would expect that Alg. 1 also gives a good approximation wrt 'Com-Eigs' (See Section 6 for experimental validation).

*Theorem 2: **Effectiveness of** NetShield.* Let $\mathcal{S}$ and $\tilde{\mathcal{S}}$ be the sets selected by Alg. 1 and by 'Com-Eval', respectively. Let $\Delta\lambda(\mathcal{S})$ and $\Delta\lambda(\tilde{\mathcal{S}})$ be the corresponding eigen-drops. Then, $\Delta\lambda(\mathcal{S}) \geq (1 - 1/e)\Delta\lambda(\tilde{\mathcal{S}})$.

Proof: Let $\mathcal{I}, \mathcal{J}, \mathcal{K}$ be three sets and $\mathcal{I} \subseteq \mathcal{J}$. Define the following three sets based on $\mathcal{I}, \mathcal{J}, \mathcal{K}$: $\mathcal{S} = \mathcal{I} \cup \mathcal{K}$, $\mathcal{T} = \mathcal{J} \cup \mathcal{K}$, $\mathcal{R} = \mathcal{J} \setminus \mathcal{I}$.

Substituting eq.(2), we have

$$\text{Sv}(\mathcal{S}) - \text{Sv}(\mathcal{I}) = \sum_{i \in \mathcal{K}} 2\lambda\mathbf{u}(i)^2 - \sum_{i,j \in \mathcal{K}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j)$$
$$- 2 \sum_{j \in \mathcal{I}, i \in \mathcal{K}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j)$$
$$\text{Sv}(\mathcal{T}) - \text{Sv}(\mathcal{J}) = \sum_{i \in \mathcal{K}} 2\lambda\mathbf{u}(i)^2 - \sum_{i,j \in \mathcal{K}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j)$$
$$- 2 \sum_{j \in \mathcal{J}, i \in \mathcal{K}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j) \quad (9)$$

According to Perron-Frobenius theorem, we have $\mathbf{u}(i) \geq$

$0 (i = 1, ..., n)$. Therefore,

$$(\text{Sv}(\mathcal{S}) - \text{Sv}(\mathcal{I})) - (\text{Sv}(\mathcal{T}) - \text{Sv}(\mathcal{J}))$$
$$= 2 \sum_{i \in \mathcal{K}, j \in \mathcal{R}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j) \geq 0 \quad (10)$$
$$\Rightarrow \text{Sv}(\mathcal{S}) - \text{Sv}(\mathcal{I}) \geq \text{Sv}(\mathcal{T}) - \text{Sv}(\mathcal{J})$$

Therefore, the function $\text{Sv}(\mathcal{S})$ is sub-modular.

Next, we can verify that node i selected in step 16 of Alg. 1 satisfies $i = argmax_{j \notin \mathcal{S}} \text{Sv}(\mathcal{S} \cup j)$ for a fixed set $\mathcal{S}$.

Next, we prove that $\text{Sv}(\mathcal{S})$ is monotonically non-decreasing wrt $\mathcal{S}$. According to eq. (9), we have

$$\text{Sv}(\mathcal{S}) - \text{Sv}(\mathcal{I}) = \sum_{i \in \mathcal{K}} 2\lambda\mathbf{u}(i)^2 - \sum_{i,j \in \mathcal{K}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j)$$
$$- 2 \sum_{j \in \mathcal{I}, i \in \mathcal{K}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j)$$
$$\geq \sum_{i \in \mathcal{K}} 2\lambda\mathbf{u}(i)^2 - 2 \sum_{j \in \mathcal{S}, i \in \mathcal{K}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j)$$
$$= 2 \sum_{i \in \mathcal{K}} \mathbf{u}(i)(\lambda\mathbf{u}(i) - \sum_{j \in \mathcal{S}} \mathbf{A}(i,j)\mathbf{u}(j))$$
$$\geq 2 \sum_{i \in \mathcal{K}} \mathbf{u}(i)(\lambda\mathbf{u}(i) - \sum_{j=1}^{n} \mathbf{A}(i,j)\mathbf{u}(j))$$
$$= 2 \sum_{i \in \mathcal{K}} \mathbf{u}(i)(\lambda\mathbf{u}(i) - \lambda\mathbf{u}(i)) = 0 \quad (11)$$

where the last equality is due to the definition of eigenvalue.

Finally, it is easy to verify that $\text{Sv}(\phi) = 0$, where ϕ is an empty set. Using the property of sub-modular functions [30], we have $\Delta\lambda(\mathcal{S}) \geq (1 - 1/e)\Delta\lambda(\tilde{\mathcal{S}})$.□

According to Lemma 2, the computational complexity of Alg. 1 is $O(nk^2 + m)$, which is much faster than both 'Com-Eigs' ($O(\binom{n}{k} \cdot m)$) and 'Com-Eval' ($O(\binom{n}{k} \cdot k^2)$).

*Lemma 2: **Computational Complexity of** NetShield.* The computational complexity of Alg. 1 is $O(nk^2 + m)$.
Proof:

The cost of step 1 is $O(m)$, and the cost of step 2 is constant. For steps 3-5, its cost is $O(n)$. For each inner loop of steps 6-17, its cost is $O(n) + O(n \cdot iter)$. Therefore, we have

$$\text{cost}(Netshield) = O(m) + O(n) + \sum_{iter=1}^{k} (n + n \cdot iter)$$
$$= O(nk^2 + m) \quad (12)$$

which completes the proof. □

Finally, according to Lemma 3, the space cost of Alg. 1 is also efficient (i.e., linear wrt the size of the graph).

*Lemma 3: **Space Cost of** NetShield.* The space cost of Alg. 1 is $O(n + m + k)$.
Proof:

The space cost of step 1 is $O(n + m + 1)$: $O(m)$ for storing the graph, $O(n + m)$ for running the eigen-decomposition algorithm, $O(n)$ for storing $\mathbf{u}$ and $O(1)$ for storing λ. The cost for step 2 is $O(1)$. For steps 3-5, we need an additional $O(n)$ space. Then, it takes $O(n)$ space for each inner loop (steps 6-17) and we can re-use this

space for the next iteration. Finally, we need $O(k)$ to store the selected nodes (step 18).

Putting the above together and ignoring the constant term, we have that the space cost of Alg. 1 is $O(n+m+k)$, which completes the proof. □

5.4 A Variant: *NetShield+* Algorithm

Recall in Lemma 1, the eigen-gap δ, max degree d and k should satisfy $\delta \geq 2\sqrt{2kd}$. Given the fact that $\lambda \leq d$, we have $\delta \leq d$. Therefore we get the constraint between max degree d and k, which can be simplified as $k \leq d/8$. The constraint implies that in order to get a good approximation of $\Delta\lambda(\mathcal{S})$ with $\mathrm{Sv}(\mathcal{S})$, the number of nodes we select to immunize should be less than $d/8$, which might not hold when the max degree of the graph is relatively small. To address this problem and further balance the optimization quality and the computational cost, we propose *NetShield+* algorithm, which is given in Alg. 2. Instead of finding out all the k nodes to delete in one round as in *NetShield* (i.e. compute the first eigenvalue and corresponding eigenvector only once), *NetShield+* tries to find out those k nodes iteratively. By fixing a batch number b as an extra input, *NetShield+* would pick out and delete b best nodes for current graph at each round, and then use the updated graph for next round of computation. More discussion on choosing an appropriate value of b is in Section 6. In Alg. 2, an extra variable b is provided as input compared to *NetShield*. It first computes the number of iterations t in step 1. In each iteration of steps 3-8, we find b nodes to delete from current graph by *NetShield* algorithm and add them to $\mathcal{S}$. At the end of each iteration, we update matrix A by deleting those selected nodes from it. The algorithm will terminate when all the k nodes are collected.

By a similar procedure for Lemma 2, we can show that the time complexity of *NetShield+* is $O(mk/b + nkb)$; and its space cost is the same as that of *NetShield*. Thus, it is still a linear algorithm wrt the size of the input graph.

Algorithm 2 *NetShield+*

Input: the adjacency matrix **A**, two integers k and b
Output: a set $\mathcal{S}$ with k nodes
1: compute the number of iterations $t = \lfloor k/b \rfloor$;
2: initialize $\mathcal{S}$ to be empty;
3: **for** $j = 1$ to t **do**
4:    initialize $\mathcal{S}'$ to be empty;
5:    $\mathcal{S}' = NetShield(A, b)$;
6:    $\mathcal{S} = \mathcal{S} \bigcup \mathcal{S}'$;
7:    update A by deleting the nodes in $\mathcal{S}'$;
8: **end for**
9: **if** $k > tb$ **then**
10:    $\mathcal{S}' = NetShield(A, k - tb)$;
11:    $\mathcal{S} = \mathcal{S} \bigcup \mathcal{S}'$;
12: **end if**
13: return $\mathcal{S}$.

---

6 EXPERIMENTAL EVALUATIONS

We present detailed experimental results in this section. All the experiments are designed to answer the following questions:

1: (*Effectiveness*) How effective is the proposed $\mathrm{Sv}(\mathcal{S})$ in real graphs?
2: (*Efficiency*) How fast and scalable is the proposed *NetShield*?

6.1 Data sets

TABLE 2: Summary of the data sets

Name	n	m
Karate	34	152
AA	418,236	2,753,798
NetFlix	2,667,199	171,460,874
Oregon-A	633	1,086
Oregon-B	1,503	2,810
Oregon-C	2,504	4,723
Oregon-D	2,854	4,932
Oregon-E	3,995	7,710
Oregon-F	5,296	10,097
Oregon-G	7,352	15,665
Oregon-H	10,860	23,409
Oregon-I	13,947	30,584

The real data sets we used are summarized in table 2. The first data set (*Karate*) is a unipartite graph, which describes the friendship among the 34 members of a karate club at a US university [65]. Each node is a member in the karate club and the existence of the edge indicates that the two corresponding members are friends. Overall, we have $n = 34$ nodes and $m = 156$ edges.

The second data set (*AA*) is an author-author network from DBLP.[4] *AA* is a co-authorship network, where each node is an author and the existence of an edge indicates the co-authorship between the two corresponding persons. Overall, we have $n = 418,236$ nodes and $m = 2,753,798$ edges. We also construct much smaller co-authorship networks, using the authors from only one conference (e.g., *KDD, SIGIR, SIGMOD*, etc.). For example, *KDD* is the co-authorship network for the authors in the 'KDD' conference. For these smaller co-authorship networks, they typically have a few thousand nodes and up to a few ten thousand edges. In this graph, the Node Immunization algorithm can help us identify a set of authors who are most important in terms of their influence in data mining and information retrieval area.

The third data set (*NetFlix*) is from the Netflix prize.[5] This is also a bipartite graph. We have two types of nodes: user and movie. The existence of an edge indicates that the corresponding user has rated the corresponding movie. Overall, we have $n = 2,667,199$ nodes and $m = 171,460,874$ edges. This is a bipartite graph, and

---

4. http://www.informatik.uni-trier.de/~ley/db/
5. http://www.netflixprize.com/

we convert it to a unipartite graph $\mathbf{A}$: $\mathbf{A} = \begin{pmatrix} \mathbf{0} & \mathbf{B} \\ \mathbf{B}' & \mathbf{0} \end{pmatrix}$, where $\mathbf{0}$ is a matrix with all zero entries and $\mathbf{B}$ is the adjacency matrix of the bipartite graph. Like the *AA* data set, by our Node Immunization algorithm, it aims to a set of well connected users/movies.

The last is a series of data set (*Oregon*) from Oregon AS (Autonomous System) router graphs, which are AS-level connectivity networks inferred from Oregon route-views [57]. The number of nodes in this set ranges from 633 to 13,947, the corresponding edges ranges from 1,086 to 30,584. The result returned by Node Immunization algorithm would be a set of most important routers in the network to immunize when virus begins to spread around the Internet.

Repeatability of Experimental Results. The code for the proposed *NetShield* and *NetShield+* is available in https://www.dropbox.com/s/aaq5ly4mcxhijmg/Netshieldplus.tar.

6.2 Effectiveness

Here, we first test the approximation accuracy of the proposed $Sv(\mathcal{S})$. Then, we compare different immunization policies, followed by some case studies. Notice that the experiment results of quality vs. speed trade-off for the proposed *NetShield*, *NetShield+*, the optimal 'Com-Eigs' and the alternative greedy method are presented in subsection 6.3.

6.2.1 Approximation quality of $Sv(\mathcal{S})$

The proposed *NetShield* is based on eq. (2). That is, we want to approximate the first eigenvalue of the perturbed matrix by λ and $\mathbf{u}$. By Lemma 1, it says that $Sv(\mathcal{S})$ is a good approximation for the actual eigen-drop $\Delta\lambda(\mathcal{S})$. Here, let us experimentally evaluate how good this approximation is on real graphs. We construct an authorship network from one of the following conferences: *'KDD', 'ICDM', 'SDM', 'SIGMOD', 'VLDB', 'NIPS', 'UAI', 'SIGIR'* and *'WWW'*. We then compute the linear correlation coefficient between $\Delta\lambda(\mathcal{S})$ and $Sv(\mathcal{S})$ with several different k values ($k = 1, 2, 5, 10, 20$). The results are shown in table 3. It can be seen that the approximation is very good - in all the cases, the linear correlation coefficient is greater than 0.9. Figure 3 gives the scatter plot of $\Delta\lambda(\mathcal{S})$ (i.e., the actual eigen-drop) vs. $Sv(\mathcal{S})$ (i.e., the proposed '*Shield-value*') for $k = 5$ on 'ICDM' data set.

6.2.2 Immunization by NetShield and NetShield+

Recall that the proposed '*Vulnerability*' score of the graph is motivated by the epidemic threshold [7]. In this paper, we primarily use SIS model (like, e.g., the flu) in our experiment for simplicity. Nonetheless, it has been proved that largest eigenvalue of the connectivity matrix can be used as epidemic threshold for many other cascade models on arbitrary networks [46].

We compare *NetShield* and *NetShield+* with the following alternative choices: (1) picking a random neighbor of a randomly chosen node [11] ('Aquaintance'), (2) picking the nodes with the highest eigen-scores $\mathbf{u}(i)(i = 1, ..., n)$
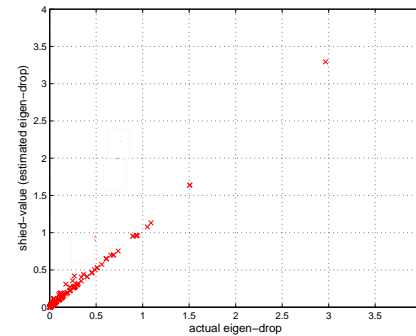

Fig. 3: *Evaluation of the approximation accuracy of $Sv(\mathcal{S})$ on the 'ICDM' graph. The proposed '*Shield-value*' $Sv(\mathcal{S})$ (y-axis) gives a good approximation for the actual eigen-drop $\Delta\lambda(\mathcal{S})$ (x-axis). Most points are on or close to the diagonal (ideal).*
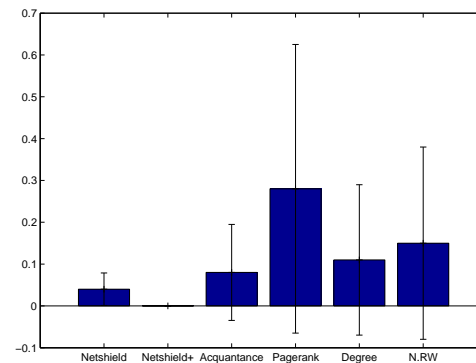

Fig. 5: *Average number of infectees at the end of each simulation using different methods and their corresponding variance*

('Eigs')[6], (3) picking the nodes with the highest abnormality scores [54] ('abnormality'), (4) picking the nodes with the highest betweenness centrality scores based on the shortest path [15] ('Short'), (5) picking the nodes with the highest betweenness centrality scores based on random walks [39] ('N.RW'), (6) picking the nodes with the highest degrees ('Degree'), (7) picking the nodes with the highest PageRank scores [43] ('PageRank') and (8) picking the nodes with highest robustness scores [8] ('Robust'). For each method, we delete 5 nodes for immunization. Let $s = \lambda \cdot b/d$ be the normalized virus strength (bigger s means more stronger virus), where b and d are the infection rate and host-recovery rate, respectively. The result is presented in figure 4, which is averaged over 100 runs. It can be seen that the proposed *NetShield+* and *NetShield* are always the best, - their curves are always the lowest which means that we always have the least number of infected nodes in the graph with this immunization strategy. Notice that the

---

6. For the un-directed graph which we focus on in this paper, 'Eigs' is equivalent to 'HITS'[29].

TABLE 3: Evaluation on the approximation accuracy of $Sv(\mathcal{S})$. Larger is better.

k	'KDD'	'ICDM'	'SDM'	'SIGMOD'	'VLDB'	'NIPS'	'UAI'	'SIGIR'	'WWW'
1	0.9519	0.9908	0.9995	1.0000	0.9548	0.9915	0.9990	0.9882	0.9438
2	0.9629	0.9910	0.9984	0.9927	0.9541	0.9914	0.9988	0.9673	0.9427
5	0.9721	0.9888	0.9992	0.9895	0.9671	0.9925	0.9987	0.9423	0.9406
10	0.9726	0.9863	0.9987	0.9852	0.9382	0.9924	0.9986	0.9327	0.9377
20	0.9683	0.9798	0.9929	0.9772	0.9298	0.9907	0.9985	0.9354	0.9288

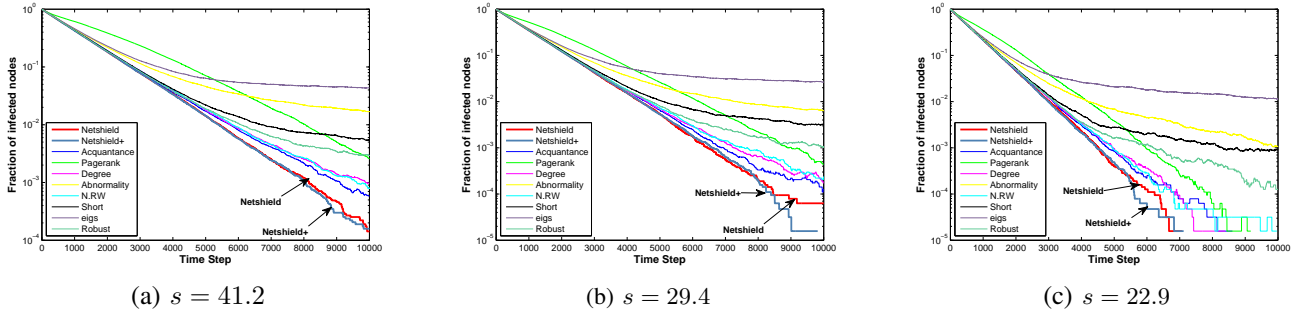(a) $s = 41.2$ (b) $s = 29.4$ (c) $s = 22.9$

Fig. 4: *Evaluation of immunization of* NetShield *and* NetShield+ *on the* Oregon-A *graph. The fraction of infected nodes (in log-scale) vs. the time step. s is normalized virus strength. Lower is better. The proposed* NetShield *and* NetShield+ *is always the best, leading to the fastest healing of the graph. Best viewed in color.*

performance of 'Eigs' is much worse than the proposed *NetShield*. This indicates that by *collectively* finding a set of nodes with the highest '*Shield-value*', we indeed obtain extra performance gain (compared with naïvely choosing the top-k nodes which have the highest *individual* '*Shield-value*' scores). Figure 5 shows the statistical significance of our proposed method on *Oregon-A* with the same setting as figure 4(b). Note that as the average numbers of infectees and variances of abnormality, shortest path betweenness centrality, eigen-scores and robustness score based methods are relative large and beyond the scope, we choose to only report the results of other stable methods. The results on other *Oregon* graphs are similar to those of *Oregon-A*.

6.2.3 Case studies

Next, we will show some case studies to illustrate the effectiveness of the proposed $Sv(\mathcal{S})$, the '*Shield-value*' score of a subset of nodes.

We run the proposed *NetShield* on *AA* data set and return the best $k = 200$ authors. Some representative authors, to name a few, are '*Sudhakar M. Reddy*', '*Wei Wang*', '*Heinrich Niemann*', '*Srimat T. Chakradhar*', '*Philip S. Yu*', '*Lei Zhang*', '*Wei Li*', '*Jiawei Han*', '*Srinivasan Parthasarathy*', '*Srivaths Ravi*', '*Antonis M. Paschalis*', '*Mohammed Javeed Zaki*', '*Lei Li*', '*Dimitris Gizopoulos*', '*Alberto L. Sangiovanni-Vincentelli*', '*Narayanan Vijaykrishnan*', '*Jason Cong*', '*Thomas S. Huang*', *etc.* We can make some very interesting observations from the result:

1 There are some multi-disciplinary people in the result. For example, Prof. Alberto L. Sangiovanni-Vincentelli from UC Berkeley is interested in 'design technology', 'cad', 'embedded systems', and 'formal verification'; Prof. Philip S. Yu from UIC is interested in

'databases', 'performance', 'distributed systems' and 'data mining'.

2 Some people show up because they are famous in one specific area, and occasionally have one/two papers in a remotely related area (therefore, increasing the path capacity between two remote areas). For example, Dr. Srimat T. Chakradhar mainly focuses on 'cad'. But he has co-authored in a 'NIPS' paper. Therefore, he creates a critical connection between these two (originally) remote areas: 'cad' and 'machine learning'.

3 Some people show up because they have ambiguous names (e.g., Wei Wang, Lei Li, Lei Zhang, Wei Li, etc.). Take 'Wei Wang' as an example; according to DBLP,[7] there are 49 different 'Wei Wang's. In our experiment, we treat all of them as one person. That is to say, it is equivalent to putting an artificial 'Wei Wang' in the graph who brings 49 different 'Wei Wang's together. These 49 'Wei Wang's are in fact spread out in quite different areas. (e.g., Wei Wang@UNC is in 'data mining' and 'bio'; Wei Wang@NUS is in 'communication'; Wei Wang@MIT is in 'non-linear systems'.)

6.3 Efficiency

We will study the wall-clock running time of the proposed *NetShield* and *NetShield+* here. Basically, we want to answer the following three questions:

1. *(Speed)* What is the speedup of the proposed *NetShield* over the straightforward methods ('Com-Eigs' and 'Com-Eval')?
2. *(Scalability)* How does *NetShield* scale with the size of the graph (n and m) and k?

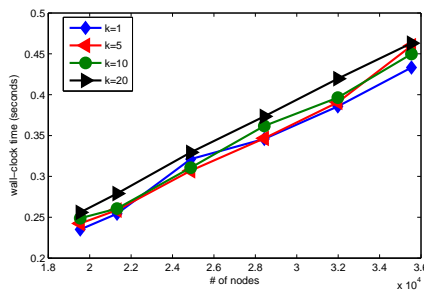7. http://www.informatik.uni-trier.de/~ey/db/indices/a-tree/w/Wang:Wei.html

3. *(Quality/Speed Trade-Off)* How does *NetShield* and *NetShield+* balance between the quality and the speed?
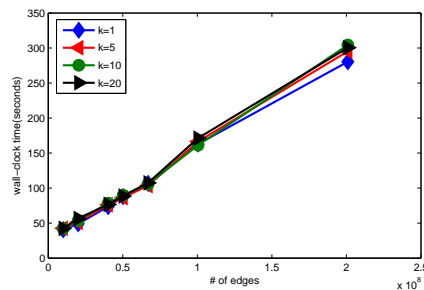
For the results we report in this subsection, all of the experiments are done on the same machine with four 2.4GHz AMD CPUs and 48GB memory, running Linux (2.6 kernel). If the program takes more than 1,000,000 seconds, we stop running it.

First, we compare *NetShield* with 'Com-Eigs' and 'Com-Eval'. Figure 6 shows the comparison on three real data sets. We can make the following conclusions: (1) Straight-forward methods ('Com-Eigs' and 'Com-Eval') are computationally intractable even for a small graph. For example, on the *Karate* data set with only 34 nodes, it takes more than 100,000 and 1,000 seconds to find the best-10 by 'Com-Eigs' and by 'Com-Eval', respectively. (2) The speedup of the proposed *NetShield* over both 'Com-Eigs' and 'Com-Eval' is huge - in most cases, we achieve *several (up to 7) orders of magnitude* speedups! (3) The speedup of the proposed *NetShield* over both 'Com-Eigs' and 'Com-Eval' quickly increases wrt the size of the graph as well as k. (4) For a given size of the graph (fixed n and m), the wall-clock time is almost constant - suggesting that *NetShield* spends most of its running time in computing λ and $\mathbf{u}$.

Next, we evaluate the scalability of *NetShield*. From figure 7, it can be seen that *NetShield* scales linearly wrt both n and m, which means that it is suitable for large graphs.

(a) changing n (fix $m = 119, 460$)

(b) changing m (fix $n = 2, 667, 119$)

Fig. 7: *Evaluation of the scalability of the proposed* Net-Shield *wrt.* n *(number of nodes) and* m *(number of edges), respectively. The wall-clock time of our* NetShield *scales linearly wrt* n *and* m.

Then, we evaluate how the proposed *NetShield* balances between the quality and speed. For the *Karate* graph, we

use the proposed *NetShield* to find a set of k nodes and check the corresponding eigen-drop (i.e., the decrease of the first eigenvalue of the adjacency matrix) as well as the corresponding wall-clock time. We compare it with 'Com-Eigs', which always gives the optimal solutions (i.e., it returns the subset that leads to the largest eigen-drop). The results (eigen-drop vs. wall-clock time) are plotted in figure 8. It can been seen that *NetShield* gains significant of speedup over the 'Com-Eigs', at the cost of a small fraction of quality loss (i.e., the green dash lines are near-flat).
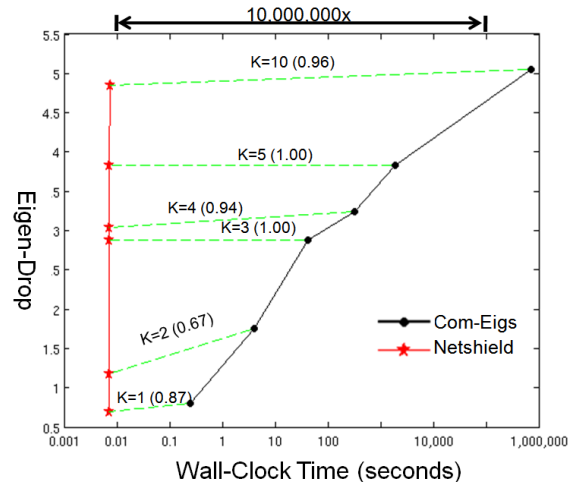

Fig. 8: *Evaluation of the quality/speed trade off. Eigen-drop vs. wall-clock time, with different budget* k. *The proposed* NetShield *(red star) achieves a good balance between eigen-drop and speed. Note that the x-axis (wall-clock time) is in logarithmic scale. The number inside the parenthesis above each green dash curve is the ratio of eigen-drop between* NetShield *and 'Com-Eigs'.* NetShield *is optimal when this ratio is 1. Best viewed in color.*

We also compare the proposed *NetShield* with the following heuristic (referred to as 'Greedy'): at each iteration, we re-compute the first eigenvector of the *current* graph and pick a node with the highest eigen-score $\mathbf{u}(i)$; then we delete this node from the graph and go to the next iteration. For the *NetFlix* graph, we find a set of k nodes and check the corresponding eigen-drop as well as the corresponding wall-clock time. The quality/speed trade-off curve is plotted in figure 9. From the figure, we can make two observations: (1) the quality of the two methods ('Greedy' vs. the proposed *NetShield*) are almost the same (note that the green dash curves in the plots are always straight flat); (2) the proposed *NetShield* is always faster than 'Greedy' (up to 103x speedup).

Finally, we evaluate how *NetShield+* further balances between the quality and speed. To try different batch value b, we move the experiment on a larger data set, *Oregon-G*. In figure 10(a), we set k to different values. For each setting of k, we change the value of b and report the relationship between ratio b/k wrt eigen-drop. The three lines all begins with $b = 1$, that is $b/k = 0.02, 0.01, 0.005$ for $k = 50, 100, 200$ respectively. Note that when b/k
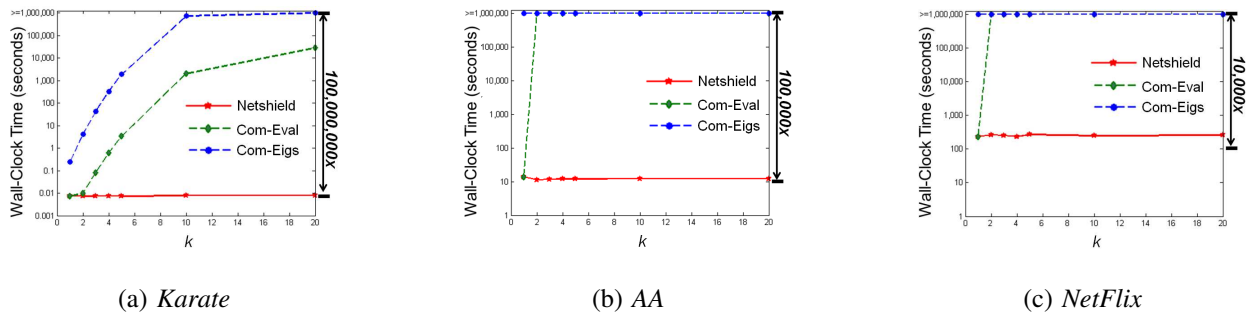
| (a) *Karate* | (b) *AA* | (c) *NetFlix* |

Fig. 6: *Wall-clock time vs. the budget k for different methods. The time is in the logarithmic scale. Our* NetShield *(red star) is much faster. Lower is better.*

increases to 1, *NetShield+* is reduced to *NetShield*. As we can see, as b increases, eigen-drop decreases, but does not make significant differences in each setting. Figure 10(b) reports the relationship between wall clock time and eigen-drop when setting different b/k ratios and k values. Setting $b = 1$ is very time consuming in all three cases. However when b is increased to $k/10$, the time is significantly reduced while eigen-drop still keeps relatively high.
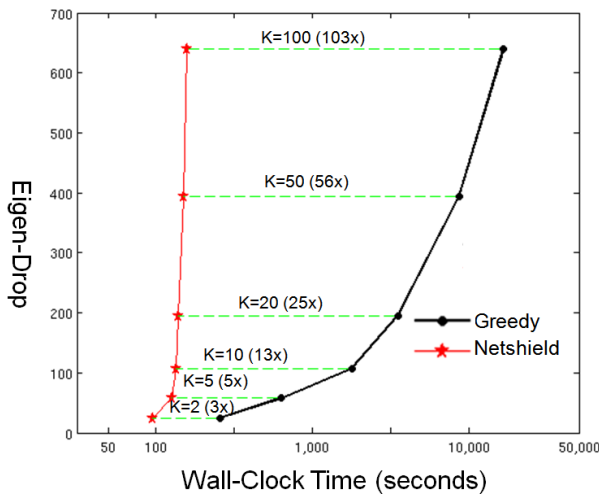

Fig. 9: *Comparison of* NetShield *vs. 'Greedy'. The proposed* NetShield *(red star) is better than 'Greedy' (i.e., faster, with the same quality). Note that the x-axis (wall-clock time) is in logarithmic scale. The number inside the parenthesis above each green dash curve is the speedup of the proposed* NetShield *over 'Greedy'. Best viewed in color.*

7 RELATED WORK

In this section, we review the related work, which can be categorized into 5 parts: measuring the importance of nodes on graphs, immunization, spectral graph analysis, influence maximization, and general graph mining.

Measuring Importance of Nodes on Graphs. In the literature, there are a lot of node importance measurements, including betweenness centrality, both the one based on the shortest path [15] and the one based on random walks [39], PageRank [43], HITS [29], and coreness score (defined by k-core decomposition) [36]. Other remotely related works include the abnormality score of a given node [54], articulation points [20], and k-vertex cut [20]. Our '*Shield-value*' score is *fundamentally* different from these node importance scores, in the sense that they *all* aim to measure the importance of an individual node; whereas our '*Shield-value*' tries to *collectively* measure the importance of a set of k nodes. Despite the fact that all these existing measures are successful for the goal they were originally designed for, they are not designed for the purpose of immunization. Therefore, it is not surprising that they lead to sub-optimal immunization results (See figure 4). Moreover, several of these importance measurements do not scale up well for large graphs, being cubic or quadratic wrt the number of nodes n, even if we use approximations (e.g., [37]). In contrast, the proposed *NetShield* is linear wrt the number of edges and the number of nodes ($O(nk^2 + m)$). Another remotely related work is outbreak detection [31] in the sense that both works aim to select a subset of "*important*" nodes on graphs. However, the motivating applications (e.g., immunization) of this work is *different* from detecting outbreak [31] (e.g., contaminants in water distribution network). Consequently we solve a *different* optimization problem (i.e., maximize the '*Shield-value*' in eq. (2)) in this paper.

Another related topic is information diffusion. Many works in this domain are based on finding out the most influential or critical nodes among the network to maximize/minimize the spread of information as shown in [60], [22], [3]. Saito et al. [49] and Yamagishi et al. [64] give the diffusion probability model and opinion formation model respectively based on node attributes. Tuli et al. [58] present an approach for selecting critical nodes for both simple and complex contagions, with the assumption that a node can contract a contagion from more than one neighbor. Another interesting work is about selecting critical nodes from the network within certain budget as in [42] and [41].

Immunization. There is vast literature on virus prop-

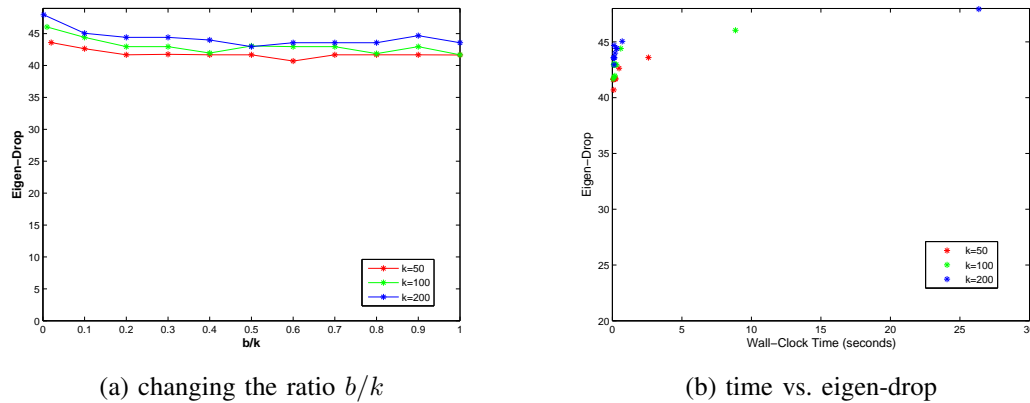(a) changing the ratio b/k (b) time vs. eigen-drop

Fig. 10: *Evaluation of quality/speed trade off of* NetShield+. *Eigen-drop does not change linearly wrt computation time, it is easy to find compromise points where we can get considerable eigen-drop with short computation time.*

agation and epidemic thresholds: for full cliques (eg., Hethcote [24]), for power-law graphs [5], and studies of heuristics for immunization policies [11]. The only papers that study *arbitrary* graphs focus on the epidemic threshold (Wang et al. [61] and its follow-up work [16], [7], [46]). In short, none of the above papers solves the problem of optimal immunization for an arbitrary, given graph.

Tong et al. in [57] address the problem of optimizing the leading eigenvalue by *edge* manipulation. Kim et al. [45] present an immunization approach of online networks based on self-similar selection, which does not require information about network morphology at individual node level. The reverse engineering of immunization problems can be defined as follows: given a snapshot of a graph in which an infection has been spreading for some time, find out the original seed set where the infection started. Related works about this topic are shown in [47] and [48]. Other related works include [44], [46], [35] and [67] which study the theory about determining epidemic in the network, algorithms about effective immunization, reverse engineering and Node Immunization given uncertain data.

Spectral Graph Analysis. Pioneering works in this aspect can be traced back to Fiedler's seminal work [14]. Representative follow-up works include [50], [40], [66], [12], etc. All of these works use the eigenvectors of the graph (or the graph Laplacian) to find communities in the graph.

Influence Maximization Although Node Immunization and influence maximization all aim to find a subset of nodes to affect the influence spread in the graph, they are different with each other in the sense that Node Immunization tries to minimize the influence spread by changing the graph structure, while influence maximization aims to choose an optimal subset of seeds to maximize the 'infected' population. The pioneering work in influence maximization is from Kempe et al [28]. To address the NP-hardness of the problem, different efficient and scalable algorithms were proposed to approximate the optimal solution for different models [9], [10], [19], [52], [18].

General Graph Mining. In recent years, graph min-

ing is a very hot research topic. Representative works include pattern and law mining [1], [6], frequent substructure discovery [63], [26], community mining and graph partition [27], [2], proximity [55], [17], [56], bridgeness-based detection of fuzzy communities [38], the network value of a customer [13], the bridge centrality [25], graph blocker [21], the connectivity of the small world [51] and social capital [32], etc. Research about sampling in graph shows that the influential individuals in the graph can be identified by only accessing to a small portion of nodes in the network. Also, certain sample biases are beneficial for many applications [34], [33]. A large amount of work is also done on analyzing the spreading process of competing information, virus and etc. [4], [59], [62]. The algorithm in [23] enables within-network and across-network classification with regional features of the graph.

8 CONCLUSION

We studied the node immunization problem on large real graphs. Besides the problem definitions, our main contributions can be summarized as the following three perspectives. First, we proposed a novel definition of 'Shield-value' score $Sv(\mathcal{S})$ for a set of nodes $\mathcal{S}$, by carefully using the results from the theory of matrix perturbation. Second, we proposed a *near-optimal* and *scalable* algorithm (*NetShield*) to find a set of nodes with the highest 'Shield-value' score. We further proposed its variant (*NetShield+*) to balance the optimization quality and speed. Finally, we conducted extensive experiments on several real data sets to illustrate both the effectiveness as well as the efficiency of our methods. Specifically, the proposed methods (a) give an effective immunization strategy (b) scale linearly with the size of the graph (number of edges) and (c) outperform competitors by several *orders of magnitude*.

Future work includes (1) to parallelize the current method (e.g., using Hadoop[8]) and (2) to study extensions for additional virus propagation models, like SIR [24] etc.

---

8. http://hadoop.apache.org/

REFERENCES

[1] R. Albert, H. Jeong, and A.-L. Barabasi. Diameter of the world wide web. *Nature*, (401):130–131, 1999.

[2] L. Backstrom, D. P. Huttenlocher, J. M. Kleinberg, and X. Lan. Group formation in large social networks: membership, growth, and evolution. In *KDD*, pages 44–54, 2006.

[3] T. Berger-Wolf et al. Working for influence: effect of network density and modularity on diffusion in networks. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*, pages 933–940. IEEE, 2011.

[4] A. Beutel, B. A. Prakash, R. Rosenfeld, and C. Faloutsos. Interacting viruses in networks: can both survive? In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434. ACM, 2012.

[5] L. Briesemeister, P. Lincoln, and P. Porras. Epidemic profiles and defense of scale-free networks. *WORM 2003*, Oct. 27 2003.

[6] A. Broder, R. Kumar, F. Maghoul1, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web: experiments and models. In *WWW Conf.*, 2000.

[7] D. Chakrabarti, Y. Wang, C. Wang, J. Leskovec, and C. Faloutsos. Epidemic thresholds in real networks. *ACM Transactions on Information and System Security (ACM TISSEC)*, 10(4), 2007.

[8] H. Chan, L. Akoglu, and H. Tong. Make it or break it: Manipulating robustness in large networks.

[9] W. Chen, Y. Wang, and S. Yang. Efficient influence maximization in social networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 199–208. ACM, 2009.

[10] W. Chen, Y. Yuan, and L. Zhang. Scalable influence maximization in social networks under the linear threshold model. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 88–97. IEEE, 2010.

[11] R. Cohen, S. Havlin, and D. ben Avraham. Efficient immunization strategies for computer networks and populations. *Physical Review Letters*, 91(24), Dec. 2003.

[12] C. H. Q. Ding, T. Li, and M. I. Jordan. Nonnegative matrix factorization for combinatorial optimization: Spectral clustering, graph matching, and clique finding. In *ICDM*, pages 183–192, 2008.

[13] P. Domingos and M. Richardson. Mining the network value of customers. In *KDD*, pages 57–66, 2001.

[14] M. Fiedler. Algebraic connectivity of graphs. 1973.

[15] L. C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.

[16] A. Ganesh, E. Massouli, and D. Towsley. The effect of network topology on the spread of epidemics. In *INFOCOM*, 2005.

[17] F. Geerts, H. Mannila, and E. Terzi. Relational link-based ranking. In *VLDB*, pages 552–563, 2004.

[18] A. Gionis, E. Terzi, and P. Tsaparas. Opinion maximization in social networks. In *SDM*, pages 387–395. SIAM, 2013.

[19] A. Goyal, W. Lu, and L. V. Lakshmanan. Simpath: An efficient algorithm for influence maximization under the linear threshold model. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pages 211–220. IEEE, 2011.

[20] N. H. and I. T. *Algorithmic Aspects of Graph Connectivity*. Cambridge University Press, 2008.

[21] Habiba and T. Y. Berger-Wolf. Graph theoretic measures for identifying effective blockers of spreading processesin dynamic networks. In *Proceedings of the MLG-ICML Workshop on Machine Learning on Graphs*, 2008.

[22] H. Habiba. *Critical Individuals in Dynamic Population Networks*. PhD thesis, Northwestern University, 2013.

[23] K. Henderson, B. Gallagher, L. Li, L. Akoglu, T. Eliassi-Rad, H. Tong, and C. Faloutsos. It's who you know: graph mining using recursive structural features. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 663–671. ACM, 2011.

[24] H. W. Hethcote. The mathematics of infectious diseases. *SIAM Review*, 42:599–653, 2000.

[25] W. Hwang, T. Kim, M. Ramanathan, and A. Zhang. Bridging centrality: graph mining from element level to group level. In *KDD*, pages 336–344, 2008.

[26] R. Jin, C. Wang, D. Polshakov, S. Parthasarathy, and G. Agrawal. Discovering frequent topological structures from graph datasets. In *KDD*, pages 606–611, 2005.

[27] G. Karypis and V. Kumar. Multilevel -way hypergraph partitioning. In *DAC*, pages 343–348, 1999.

[28] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. *KDD*, 2003.

[29] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. In *ACM-SIAM Symposium on Discrete Algorithms*, 1998.

[30] A. Krause and C. Guestrin. Near-optimal observation selection using submodular functions. In *AAAI*, pages 1650–1654, 2007.

[31] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. S. Glance. Cost-effective outbreak detection in networks. In *KDD*, pages 420–429, 2007.

[32] L. Licamele and L. Getoor. Social capital in friendship-event networks. In *ICDM*, pages 959–964, 2006.

[33] A. S. Maiya. *Sampling and Inference in Complex Networks*. PhD thesis, Stanford University, 2011.

[34] A. S. Maiya and T. Y. Berger-Wolf. Benefits of bias: Towards better characterization of network sampling. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 105–113. ACM, 2011.

[35] Y. Matsubara, Y. Sakurai, B. A. Prakash, L. Li, and C. Faloutsos. Rise and fall patterns of information diffusion: model and implications. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 6–14. ACM, 2012.

[36] J. Moody and D. R. White. Social cohesion and embeddedness: A hierarchical conception of social groups. *American Sociological Review*, pages 1–25, 2003.

[37] J. I. Munro and D. Wagner. Better approximation of betweenness centrality. 2008.

[38] T. Nepusz, A. Petraczi, L. Negyessy, and F. Bazso. Fuzzy communities and the concept of bridgeness in complex networks. *Physics and Society*, 2007.

[39] M. Newman. A measure of betweenness centrality based on random walks. *Social Networks*, 27:39–54, 2005.

[40] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *NIPS*, pages 849–856, 2001.

[41] H. Nguyen. *Interactions on Complex Networks: Inference Algorithms and Applications*. PhD thesis, University of Houston, 2013.

[42] H. Nguyen and R. Zheng. On budgeted influence maximization in social networks. *Selected Areas in Communications, IEEE Journal on*, 31(6):1084–1094, 2013.

[43] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998. Paper SIDL-WP-1999-0120 (version of 11/11/1999).

[44] B. A. Prakash. propagation and immunization in large networks. *XRDS: Crossroads, The ACM Magazine for Students*, 19(1):56–59, 2012.

[45] B. A. Prakash, L. Adamic, T. Iwashyna, H. Tong, and C. Faloutsos. Fractional immunization in hospital-transfer graphs.

[46] B. A. Prakash, D. Chakrabarti, N. C. Valler, M. Faloutsos, and C. Faloutsos. Threshold conditions for arbitrary cascade models on arbitrary networks. *Knowledge and information systems*, 33(3):549–575, 2012.

[47] B. A. Prakash, J. Vreeken, and C. Faloutsos. Spotting culprits in epidemics: How many and which ones? In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 11–20. IEEE, 2012.

[48] B. A. Prakash, J. Vreeken, and C. Faloutsos. Efficiently spotting the starting points of an epidemic in a large graph. *Knowledge and Information Systems*, pages 1–25, 2013.

[49] K. Saito, K. Ohara, Y. Yamagishi, M. Kimura, and H. Motoda. Learning diffusion probability based on node attributes in social networks. In *Foundations of Intelligent Systems*, pages 153–162. Springer, 2011.

[50] J. Shi and J. Malik. Normalized cuts and image segmentation. In *CVPR*, pages 731–737, 1997.

[51] X. Shi, M. Bonner, L. A. Adamic, and A. C. Gilbert. The very small world of the well-connected. In *Hypertext*, pages 61–70, 2008.

[52] Y. Singer. How to win friends and influence people, truthfully: influence maximization mechanisms for social networks. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 733–742. ACM, 2012.

[53] G. W. Stewart and J.-G. Sun. *Matrix Perturbation Theory*. Academic Press, 1990.

[54] J. Sun, H. Qu, D. Chakrabarti, and C. Faloutsos. Neighborhood formation and anomaly detection in bipartite graphs. In *ICDM*, pages 418–425, 2005.

[55] H. Tong, C. Faloutsos, and J.-Y. Pan. Fast random walk with restart and its applications. In *ICDM*, pages 613–622, 2006.

[56] H. Tong, J. He, M. Li, W.-Y. Ma, H.-J. Zhang, and C. Zhang. Manifold-ranking-based keyword propagation for image retrieval. *EURASIP Journal on Applied Signal Processing*, 2006:Article ID 79412, 10 pages, 2006. doi:10.1155/ASP/2006/79412.

[57] H. Tong, B. A. Prakash, T. Eliassi-Rad, M. Faloutsos, and C. Faloutsos. Gelling, and melting, large graphs by edge manipulation. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 245–254. ACM, 2012.

[58] G. Tuli, C. J. Kuhlman, M. V. Marathe, S. Ravi, and D. J. Rosenkrantz. Blocking complex contagions using community structure. *MAIN*, 2012.

[59] N. C. Valler. *Spreading Processes on Networks Theory and Applications*. PhD thesis, UNIVERSITY OF CALIFORNIA, 2012.

[60] D. Wang, Z. Wen, H. Tong, C.-Y. Lin, C. Song, and A.-L. Barabási. Information spreading in context. In *Proceedings of the 20th international conference on World wide web*, pages 735–744. ACM, 2011.

[61] Y. Wang, D. Chakrabarti, C. Wang, and C. Faloutsos. Epidemic spreading in real networks: An eigenvalue viewpoint. *SRDS*, 2003.

[62] X. Wei, N. C. Valler, B. A. Prakash, I. Neamtiu, M. Faloutsos, and C. Faloutsos. Competing memes propagation on networks: A network science perspective. *Selected Areas in Communications, IEEE Journal on*, 31(6):1049–1060, 2013.

[63] D. Xin, J. Han, X. Yan, and H. Cheng. Mining compressed frequent-pattern sets. In *VLDB*, pages 709–720, 2005.

[64] Y. Yamagishi, K. Saito, K. Ohara, M. Kimura, and H. Motoda. Learning attribute-weighted voter model over social networks. *Journal of Machine Learning Research-Proceedings Track*, 20:263–280, 2011.

[65] W. W. Zachary. An information flow model for conflict and fission in small groups. pages 452–473, 1977.

[66] H. Zha, X. He, C. H. Q. Ding, M. Gu, and H. D. Simon. Spectral relaxation for k-means clustering. In *NIPS*, pages 1057–1064, 2001.

[67] Y. Zhang and B. A. Prakash. Scalable vaccine distribution in large graphs given uncertain data. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 1719–1728. ACM, 2014.

APPENDIX
PROOF OF NP-COMPLETENESS OF *K*-NODE IMMUNIZATION

Proof:

We consider the decision version of the *K*-Node immunization problem as follows.

Problem 4: K-Node Immunization (Decision Version) ($IMM(G,k)$)

Given: A large un-directed un-weighted connected graph G with n nodes and an integer k;

Find: A subset $\mathcal{S}$ of k nodes. By deleting $\mathcal{S}$ from graph G (with adjacency matrix $\mathbf{A}$), we get a new graph $G^{(\mathcal{S})}$(with adjacency matrix $\hat{\mathbf{A}}$), in which $\lambda^{(\mathcal{S})} \leq \tau$. To make the problem easier, we proof that the problem is already NP-complete when $\tau = 0$.

First, we show that *K*-Node immunization problem is in NP: given subset ($\mathcal{S}$) to be deleted from graph G, we can check in poly-time if the first eigenvalue of new graph $G^{(\mathcal{S})}$ is less than 0 or not.

Second, we prove that *K*-Node immunization problem is poly-time reducible from a known NP-complete problem, i.e., the Independent Set problem($IND(G,k)$).

Problem 5: Independent Set problem($IND(G,k)$)

Given a large un-directed un-weighted connected graph $G = (V, E)$ and a number k > 0, is there a set of k vertices, no two of which are adjacent?

Assume the size of G is n. Given an instance of $IND(G,k)$, we create an instance $IMM(G, n-k)$ (delete $n - k$ nodes in G such that the the first eigenvalue in new graph is less or equal to 0). We now need to prove two things:

1. If there is a YES answer to $IND(G,k)$, then there is a YES answer to $IMM(G, n-k)$.

The adjacency matrix of G which has YES answer to $IND(G,k)$ is

$$\mathbf{A} = \begin{pmatrix} \mathbf{S}_{k \times k} & \mathbf{X}_{k \times (n-k)} \\ \mathbf{X}_{k \times (n-k)} & \mathbf{T}_{(n-k) \times (n-k)} \end{pmatrix}$$

where $\mathbf{S}_{k \times k} = \mathbf{0}$, because the k nodes in S are independent to each other. By deleting the rest $n - k$ nodes in T ($T = V/S$), we have $\mathbf{X}_{k \times (n-k)} = \mathbf{0}$, $\mathbf{T}_{(n-k) \times (n-k)} = \mathbf{0}$. Therefore the adjacency matrix for new graph $G^{(T)}$ has $\hat{\mathbf{A}} = \mathbf{0}$. Hence $\lambda^{(T)} = \lambda(\mathbf{0}) = 0$. So there is a YES answer to $IMM(G, n-k)$.

2. If there is a NO answer to $IND(G,k)$, then there is a NO answer to $IMM(G, n-k)$.

Suppose we have a YES answer to $IMM(G, n-k)$. Then by deleting $n - k$ nodes from graph G (suppose they are in T), we will get new graph $G^{(T)}$ with $\lambda^{(T)} \leq 0$ where

$$\hat{\mathbf{A}} = \begin{pmatrix} \mathbf{S}_{k \times k} & \mathbf{0}_{k \times (n-k)} \\ \mathbf{0}_{k \times (n-k)} & \mathbf{0}_{(n-k) \times (n-k)} \end{pmatrix}$$

Since $\mathbf{S}_{k \times k} \geq \mathbf{0}$, to satisfy $\lambda^{(T)} \leq 0$, we need to have $\mathbf{S}_{k \times k} = \mathbf{0}$, which implies that all the k nodes in S are independent to each other. The conclusion is contradict with the assumption that there is a NO answer to $IND(G,k)$, therefore $IMM(G, n-k)$ can only have NO answer here.

Hence *K*-node Immunization (Decision Version) is NP-complete.$\square$