On the Vulnerability of Large Graphs

Hanghang Tong, B. Aditya Prakash, Charalampos Tsourakakis,
Tina Eliassi-Rad, Christos Faloutsos, Polo 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? We need (a) a measure of the '*Vulnerability*' of a given network, (b) a measure of the '*Shield-value*' of a specific set of k nodes and (c) a fast algorithm to choose the best such k nodes.*

We answer all these three questions: we give the justification behind our choices, we show that they agree with intuition as well as recent results in immunology. Moreover, we propose NetShield*, a fast and scalable algorithm which is provably* near-optimal *(within constant factor from optimal). Finally, we give experiments on large real graphs, where* NetShield *achieves tremendous speed savings exceeding* 7 orders of magnitude*, against straightforward competitors.*

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.

The problem boils down to three questions, which we address here:

Q1. *Vulnerability measure:* How to capture the '*Vulnerability*' of the graph, in a single number? That is, how likely/easily that a graph will be infected by a virus.

Q2. '*Shield-value*'*:* How 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 immunized/removed?

Q3. *Algorithm:* How to quickly determine the k nodes that *collectively* exhibit the highest '*Shield-value*' score on large, disk-resident graphs?

We start by adopting the first[1] eigenvalue λ of the graph as the '*Vulnerability*' measurement (for Q1), which is motivated from immunology and graph loop/path capacity. Based on that, we propose a novel definition of the '*Shield-value*' score $\mathrm{Sv}(\mathcal{S})$ for a specific set of nodes (for Q2). 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 Q3). We conduct extensive experiments on several real data sets, illustrating the effectiveness and efficiency of the proposed methods. Specifically, our experiments show that the proposed method *NetShield* (a) leads an effective immunization strategy; (b) scales *linearly* with the size of the graph; and (c) is dramatically faster than competitors (over *7 orders of magnitude*).

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. We 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 eigen-value of $\mathbf{A}$, then λ^{i} is the first eigen-value of $\mathbf{A}$ after deleting its $i^{(\mathrm{th})}$

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

row/column. We use $(\lambda_i, \mathbf{u}_i)$ to denote the i^{th} 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
$\mathbf{A}, \mathbf{B}, \ldots$	matrices (bold upper case)
$\mathbf{A}(i, j)$	the element at the i^{th} row and j^{th} column of matrix $\mathbf{A}$
$\mathbf{A}(i, :)$	the i^{th} row of matrix $\mathbf{A}$
$\mathbf{A}(:, j)$	the j^{th} column of matrix $\mathbf{A}$
$\mathbf{A}'$	transpose of matrix $\mathbf{A}$
$\mathbf{a}, \mathbf{b}, \ldots$	column vectors
$\mathcal{S}, \mathcal{T}, \ldots$	sets (calligraphic)
n	number of nodes in the graph
m	number of edges in the graph
$(\lambda_i, \mathbf{u}_i)$	the i^{th} eigen-pair of $\mathbf{A}$
λ	first eigen-value of $\mathbf{A}$ (i.e., $\lambda \triangleq \lambda_1$)
$\mathbf{u}$	first eigen-vector of $\mathbf{A}$ (i.e., $\mathbf{u} \triangleq \mathbf{u}_1$)
$\lambda^{(i)}$, $\lambda^{(\mathcal{S})}$	first eigen-value 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: $\Delta\lambda(\mathcal{S}) = \lambda - \lambda^{(\mathcal{S})}$
$\mathrm{Sv}(i)$	'*Shield-value*' score of node i
$\mathrm{Sv}(\mathcal{S})$	'*Shield-value*' score of nodes in $\mathcal{S}$
$\mathrm{V}(\mathbf{G})$	'*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 $V(\mathbf{G})$, 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 $\mathbf{A}$;*

Find: *A single number $Sv(\mathcal{S})$, 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 $\mathbf{A}$ 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

Here, we focus on Problem 1. We suggest using the first eigenvalue λ as the solution. We should point out that it *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.

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

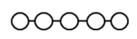
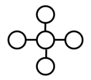
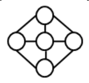
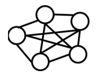
			
(a) $\lambda = 1.7$	(b)$\lambda = 2.0$	$\lambda = 2.9$	$\lambda = 4.0$

Figure 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.*

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.

Notice that the concept of '*Vulnerability*' is *different* from vertex connectivity of the graph [13]. For '*Vulnerability*', we want to quantify how likily/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 1s). But graph (b) is more vulnerable to the virus attack. Also notices 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 [5]: 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 the virus attack).

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

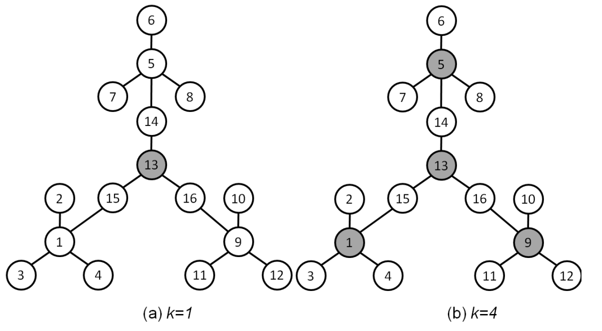


Figure 2. *An example on measuring the '*Shield-value*' score of a given set of nodes. The best k nodes found by our* NetShield *are shaded. In (a), notice that the highest degree nodes (e.g., node 1) is not chosen. In (b), immunizing the shaded nodes makes the remaining graph most robust to the virus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

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

where $Sv(\mathcal{S})$ 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 [31]), we have

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

Let $\tilde{\lambda}_i (i = 2, ..., n)$ be the corresponding perturbed eigenvalue of $\lambda_i (i = 2, ..., n)$. Again, by the matrix perturbation theory (p.203 [31]), we have

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

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,

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

which completes the proof. $\square$

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 straight-forward methods (referred to as 'Com-Eigs', and 'Com-Eval') are computationally intractable. Then, we present the proposed *NetShield* algorithm. Finally, we analyze its accuracy as well as its computational complexity.

5.1 Preliminaries

There are two obviously straight-forward 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 eigen-value. 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 eigen-value $\lambda_{\mathcal{S}}$, $\binom{n}{k}$ times, and pick the subset with the highest $\mathrm{Sv}(\mathcal{S})$. We refer to this strategy as 'Com-Eval'. Compared with the straight-forward 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!

[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.

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 $\mathcal{S}$.

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 eigen-value $\lambda$ of $\mathbf{A}$; let $\mathbf{u}$ be the corresponding eigen-vector $\mathbf{u}(j) (j = 1, ..., n)$;
2:  initialize $\mathcal{S}$ to be empty;
3:  for $j = 1$ to $n$ do
4:      $\mathbf{v}(j) = (2 \cdot \lambda - \mathbf{A}(j, j)) \cdot u(j)^2$;
5:  end for
6:  for iter = 1 to $k$ do
7:      let $\mathbf{B} = \mathbf{A}(:, \mathcal{S})$;
8:      let $\mathbf{b} = \mathbf{B} \cdot \mathbf{u}(\mathcal{S})$;
9:      for $j = 1$ to $n$ do
10:         if $j \in \mathcal{S}$ then
11:             let score$(j) = -1$;
12:         else
13:             let score$(j) = \mathbf{v}(j) - 2 \cdot \mathbf{b}(j) \cdot \mathbf{u}(j)$;
14:         end if
15:     end for
16:     let $i = \mathrm{argmax}_j$ 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 1 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 $\mathcal{I}, \mathcal{J}, \mathcal{K}$: $\mathcal{S} = \mathcal{I} \cup \mathcal{K}, \quad \mathcal{T} = \mathcal{J} \cup \mathcal{K}, \quad \mathcal{R} = \mathcal{J} \setminus \mathcal{I}$.

Substituting eq.(2), we have

$$\begin{aligned}
\mathrm{Sv}(\mathcal{S}) &- \mathrm{Sv}(\mathcal{I}) = \sum_{i\in\mathcal{K}} 2\lambda\mathbf{u}(i)^2 - \sum_{i,j\in\mathcal{K}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j) \\
&- 2\sum_{j\in\mathcal{I}, i\in\mathcal{K}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j) \\
\mathrm{Sv}(\mathcal{T}) &- \mathrm{Sv}(\mathcal{J}) = \sum_{i\in\mathcal{K}} 2\lambda\mathbf{u}(i)^2 - \sum_{i,j\in\mathcal{K}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j) \\
&- 2\sum_{j\in\mathcal{J}, i\in\mathcal{K}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j) \qquad (9)
\end{aligned}$$

According to Perron-Frobenius theorem, we have $\mathbf{u}(i) \geq 0 (i = 1, ..., n)$. Therefore,

$$\begin{aligned}
(\mathrm{Sv}(\mathcal{S}) &- \mathrm{Sv}(\mathcal{I})) - (\mathrm{Sv}(\mathcal{T}) - \mathrm{Sv}(\mathcal{J})) \\
&= 2\sum_{i\in\mathcal{K}, j\in\mathcal{R}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j) \geq 0 \qquad (10) \\
&\Rightarrow \mathrm{Sv}(\mathcal{S}) - \mathrm{Sv}(\mathcal{I}) \geq \mathrm{Sv}(\mathcal{T}) - \mathrm{Sv}(\mathcal{J})
\end{aligned}$$

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

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

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

$$\begin{aligned}
\mathrm{Sv}(\mathcal{S}) &- \mathrm{Sv}(\mathcal{I}) = \sum_{i\in\mathcal{K}} 2\lambda\mathbf{u}(i)^2 - \sum_{i,j\in\mathcal{K}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j) \\
&- 2\sum_{j\in\mathcal{I}, i\in\mathcal{K}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j) \\
&\geq \sum_{i\in\mathcal{K}} 2\lambda\mathbf{u}(i)^2 - 2\sum_{j\in\mathcal{S}, i\in\mathcal{K}} \mathbf{A}(i,j)\mathbf{u}(i)\mathbf{u}(j) \\
&= 2\sum_{i\in\mathcal{K}} \mathbf{u}(i)(\lambda\mathbf{u}(i) - \sum_{j\in\mathcal{S}} \mathbf{A}(i,j)\mathbf{u}(j)) \\
&\geq 2\sum_{i\in\mathcal{K}} \mathbf{u}(i)(\lambda\mathbf{u}(i) - \sum_{j=1}^{n} \mathbf{A}(i,j)\mathbf{u}(j)) \\
&= 2\sum_{i\in\mathcal{K}} \mathbf{u}(i)(\lambda\mathbf{u}(i) - \lambda\mathbf{u}(i)) = 0 \qquad (11)
\end{aligned}$$

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

Finally, it is easy to verify that $\mathbf{Sv}(\phi) = 0$, where ϕ is an empty set. Using the property of sub-modular functions [20], 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' ($O(\binom{n}{k}m)$) and 'Com-Eval' ($O(\binom{n}{k}k^2)$).

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

Proof: Omitted for brevity. □

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: Omitted for brevity. □

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

We used three real data sets, which 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 [36]. 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.

The last 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 $m = 171,460,874$

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

Table 3. Evaluation on the approximation accuracy of $f(\mathcal{S})$. Larger is better.

k	'KDD'	'ICDM'	'SDM'	'SIGMOD'
1	0.9519	0.9908	0.9995	1.0000
2	0.9629	0.9910	0.9984	0.9927
5	0.9721	0.9888	0.9992	0.9895
10	0.9726	0.9863	0.9987	0.9852
20	0.9683	0.9798	0.9929	0.9772

edges. This is a bipartite graph, and we convert it to a uni-partite 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.

Repeatability of Experimental Results. All the data sets we used in this paper are available on-line. After the paper is accepted, we will release the code for the proposed *NetShield*, either on the first author's website or via email-request.

6.2 Effectiveness

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

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

The proposed *NetShield* is based on eq. (2). That is, we want to approximate the first eigen-value of the perturbed matrix by λ and $\mathbf{u}$. By Lemma 1, it says that $\text{Sv}(\mathcal{S})$ 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' and 'SIGMOD'*. We then compute the linear correlation coefficient between $\Delta\lambda(\mathcal{S})$ and $\text{Sv}(\mathcal{S})$ 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 0.95. Figure 3 gives the scatter plot of $\Delta\lambda(\mathcal{S})$ (i.e., the actual eigen-drop) vs. $\text{Sv}(\mathcal{S})$ (i.e., the proposed '*Shield-value*') for $k = 5$ the on 'ICDM' data set.

6.2.2 Immunization by *NetShield*

The proposed '*Vulnerability*' score of the graph is motivated by the epidemic threshold [5]. As a consequence, the proposed *NetShield* leads to a natural immunization strategy for the SIS model (susceptible-infective-susceptible, like, e.g., the flu): quarantine or delete the subset of the nodes detected by *NetShield* in order to prevent an epidemic from

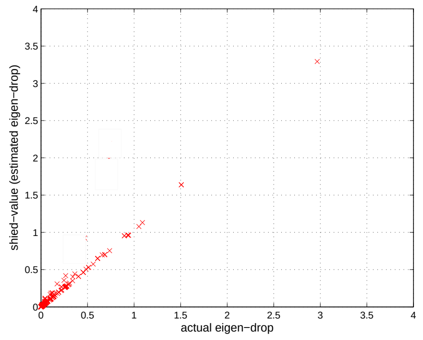


Figure 3. *Evaluation of the approximation accuracy of* $\text{Sv}(\mathcal{S})$ *on the 'ICDM' graph. The proposed* 'Shield-value' *Sv (y-axis) gives a good approximation for the actual eigen-drop* $\Delta\lambda(\mathcal{S})$ *(x-axis). Most points are on or close to the diagonal (ideal).*

breaking out. [6]

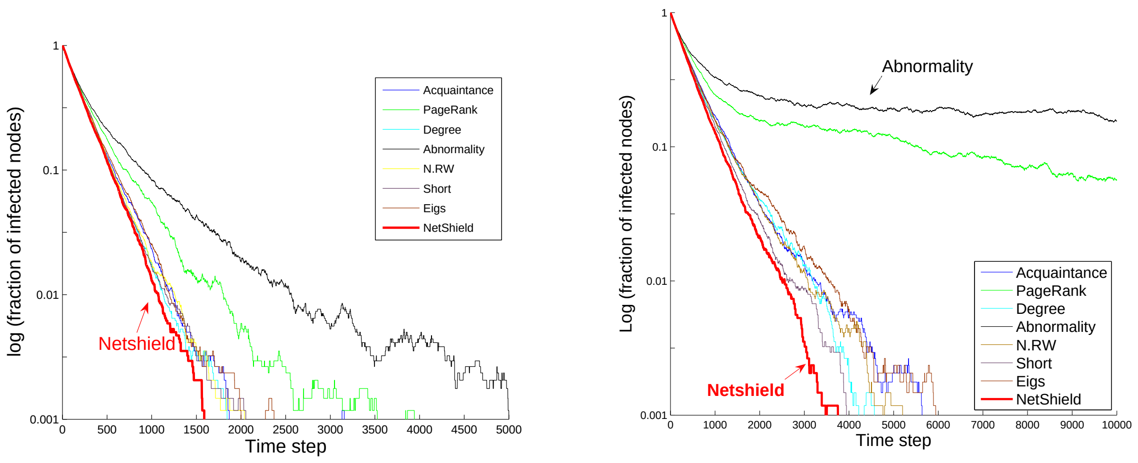
We compare it with the following alternative choices: (1) picking a random neighbor of a randomly chosen node[6] ('Aquaintance'), (2) picking the nodes with the highest eigen scores $\mathbf{u}(i)(i = 1, ..., n)$ ('Eigs')[7], (3) picking the nodes with the highest abnormality scores [32] ('abnormality'), (4) picking the nodes with the highest betweenness centrality scores based on the shortest path [10]('Short'), (5) picking the nodes with the highest betweenness centrality scores based on random walks [26]('N.RW'), (6) picking the nodes with the highest degress ('Degree'), and (7) picking the nodes with the highest PageRank scores [28]('PageRank'). 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 death rate, respectively. The result is presented in figure 4, which is averaged over 1000 runs. It can be seen that the proposed *NetShield* is always the best, - its curve is 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 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).

6.2.3 Case studies

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

[6]According to [5], for SIR (susceptible-infective-recovered) model, its epidemic threshold is also determined by λ. Therefore, we expect that our *NetShield* can also help with the immunization for the SIR model.

[7]For the un-directed graph which we focus on in this paper, 'Eigs' is equivalent to 'HITS'[19].



(a) $s = 1.4$ (b) $s = 2.9$

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

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,[8] there are 7 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 7 different 'Wei Wang's together. These 7 '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'.)

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

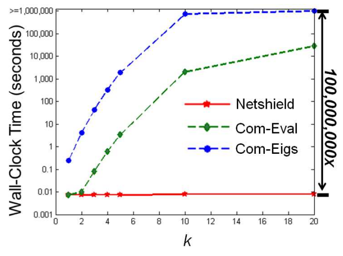
6.3 Efficiency

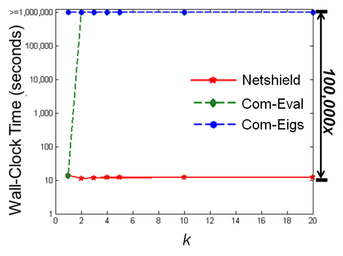
We will study the wall-clock running time of the proposed *NetShield* here. Basically, we want to answer the following three questions:

1. *(Speed)* What is the speedup of the proposed *NetShield* over the straight-forward methods ('Com-Eigs' and 'Com-Eval')?
2. *(Scalability)* How does *NetShield* scale with the size of the graph (n and m) and k?
3. *(Quality/Speed Trade-Off)* How does *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 5 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


(a) *Karate*


(b) *AA*

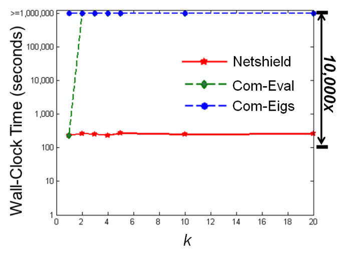
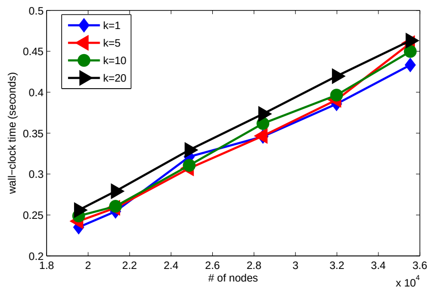

(c) *NetFlix*

Figure 5. *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.*

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 6, 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 $m = 119, 460$)

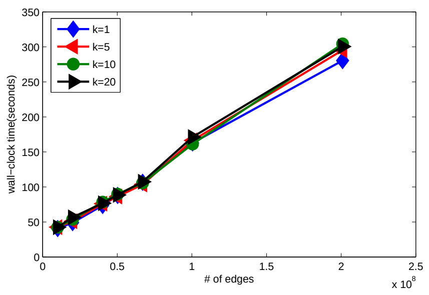

(b) changing m (fix $n = 2, 667, 119$)

Figure 6. *Evaluation of the scalability of the proposed* NetShield *wrt. n (number of nodes) and m (number of edges), respectively. The wall-clock time of our* NetShield *scales linearly wrt n and m.*

Finally, 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 eigen-value 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 7. 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).

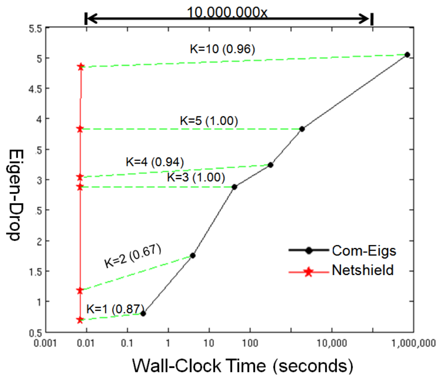


Figure 7. *Evaluation of the quality/speed trade off. Eigen-drop vs.wall-clock time, with different budget k.The proposed* NetShield *(red star) achieves a good balance between eigen-drop and speed. Note that the x-axis (wall-clock time) is in logarithmic scale. The number inside the parenthesis above 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 8. 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).

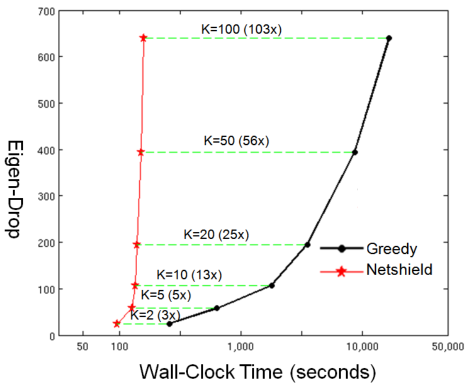


Figure 8. *Comparison of* NetShield *vs. 'Greedy'. The proposed* NetShield *(red star) is better than 'Greedy' (i.e., faster, with the same quality). Note that the x-axis (wall-clock time) is in logarithmic scale. The number inside the parenthesis above each green dash curve is the speedup of the proposed* NetShield *over 'Greedy'. Best viewed in color.*

7 Related Work

In this section, we review the related work, which can be categorized into 4 parts: measuring the importance of nodes on graphs, immunization, spectral graph analysis, and general graph mining.

Measuring Importance of Nodes on Graphs. In the literature, there are a lot of node importance measurements, including betweeness centrality, both the one based on the shortest path [10] and the one based on random walks [26], PageRank [28], HITS [19], and coreness score (defined by k-core decomposition) [23]. Other remotely related works include the abnormality score of a given node [32], articulation points [13], and k-vertex cut [13]. 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., [24]). 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 [21] 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 [21] (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.

Immunization. There is vast literature on virus propagation and epidemic thresholds: for full cliques (eg., Hethcote [15]), for power-law graphs [3], and studies of heuristics for immunization policies [6]. The only papers that study *arbitrary* graphs focus on the epidemic threshold (Wang et al. [34] and its follow-up work [11],[5].

In short, none of the above papers solves the problem of optimal immunization for an arbitrary, given graph.

Spectral Graph Analysis. Pioneering works in this aspect can be traced back to Fiedler's seminal work [9]. Representative follow-up works include [29, 27, 37, 7], etc. All of these works use the eigen-vectors of the graph (or the graph Laplacian) to find communities in the graph.

General Graph Mining. In recent years, graph mining is a very hot research topic. Representative works include pattern and law mining [1, 4], frequent substructure discovery [35, 17], community mining and graph partition [18, 2], proximity [33, 12], bridgeness-based detection of fuzzy communities [25], the network value of a customer [8], the bridge centrality [16], graph blocker [14], the connectivity of the small world [30] and social capital [22], etc.

8 Conclusion

We studied the '*Vulnerability*' of large real graphs. Besides the problem definitions, our main contributions are

1. A novel definition of '*Shield-value*' score $\mathrm{Sv}(\mathcal{S})$ for a set of nodes $\mathcal{S}$, by carefully using the results from the theory of matrix perturbation; ($\mathrm{Sv}()$ is a good approximation to the eigen-drop $\Delta\lambda$, the reduction of '*Vulnerability*', see Lemma 1).
2. A *near-optimal* and *scalable* algorithm (*NetShield*) to find a set of nodes with the highest '*Shield-value*' score, by showing that our setting has the submodularity property (i.e., Theorem 1).
3. Extensive experiments on several real data sets, illustrating both the effectiveness as well as the efficiency of our methods.

Specifically, the proposed method (a) gives an effective immunization strategy (b) scales linearly with the size of the graph (number of edges) and (c) outperforms competitors by several *orders of magnitude*.

A promising research direction is to parallelize the current method (e.g., using Hadoop[9]). Another one is to study extensions for additional virus propagation models, like SIR [15] etc.

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

[4] 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.

[5] 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.

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

[7] 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.

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

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

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

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

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

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

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

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

[16] 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.

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

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

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

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

[21] 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.

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

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

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

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

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

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

[28] 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).

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

[30] 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.

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

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

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

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

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

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

[37] 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.

[9]http://hadoop.apache.org/