2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining

Flow-Aware Vertex Protection Strategy on Large Social Networks

Arie Wahyu Wijayanto
Department of Computer Science, School of Computing
Tokyo Institute of Technology
ariewahyu@net.c.titech.ac.jp

Tsuyoshi Murata
Department of Computer Science, School of Computing
Tokyo Institute of Technology

***Abstract*—Given a large graph, such as social network, how to determine set of vertices should we protect given the k budget such that the percentage of vertices that remain uninfected at the end of infection propagation is maximized? Considering the intricacy of this problem and the requirement of scalability, the existing methods are not scalable. On the other hand, the connections among vertices in many real world contagions are usually not solely binary entities (either present or not) but have associated magnitudes and directions. We formulate the flow-aware vertex protection (FAVP) problem to elaborate more efficient and realistic way to prevent infection spreading in graphs by protecting a set of vertices. We also demonstrate that the FAVP problem is NP-Hard. Finally, we propose an efficient and scalable algorithm, called GraphShield by taking into account the role of infection flow, graph connectivity, and outdegree centrality. Experimental results on many real network datasets show that the GraphShield outperforms the state-of-the-art algorithms regarding both effectiveness and efficiency.**

***Index Terms*—network analysis, graph mining, large graph**

I. Introduction

Inhibiting infection spreading is an important concern in many different real-world network application. For example, in the presence of fake news (or rumor, or spam) spreading in a social network which units should we protect to prevent a further spreading? How to find the best approach to determine which set of network units should we protect to make the network more robust against the random spreading failures (or viruses, or diseases)? These problems have already known as vertex protection problem in a network [1]–[4].

The structure of networks dictates how quickly the infection will propagate. We exploit this benefit to determine specific vertices for protection strategy, such that the infection spreading is considerably diminished. There are traditional approaches of node importance in graphs such as degree centrality [5], PageRank centrality [6], closeness centrality [7], etc. However, they are not designed specifically for infection spreading case [1], [5]. Thus, there is no special attention to scalability, which could lead to difficult implementation in large graphs.

ASONAM '17, July 31 - August 03, 2017, Sydney, Australia

ACM ISBN 978-1-4503-4993-2/17/07/$15.00
http://dx.doi.org/10.1145/3110025.3110033

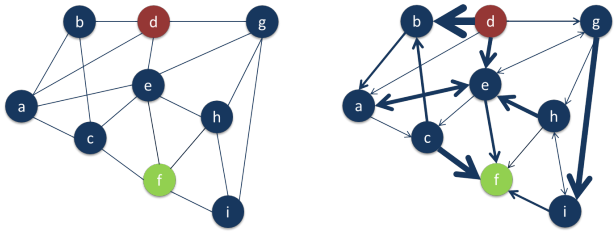


Figure 1: **Problem representation**. The arrow in the right image indicates *followed by*, while the magnitude represents the probability of getting influenced or infected due to their frequency of interaction, filtering status, etc. *Red* and *green* nodes have the same degree of connection but have different possibility to infect their connected counterparts.

Existing recent methods in vertex protection problem have mostly concentrated on a general yet simplified presumption and ignored the role of degree centrality [1], [2], [4]. Chen et al. [1] investigated this problem using the elaboration of matrix perturbation theory and proposed the near-optimal sub-modular algorithms called NetShield and NetShield+. They presented the utilization of perturbed matrix characteristics to define a sub-modular protection measurement of a particular set of vertices. They noticed that the largest eigenvalue represents the connectivity and vulnerability of a particular graph. Thus, the set of vertices having a maximum drop in eigenvalue regarding their deletion from the graph, have more likely to be protected.

However, ignoring the direction and flow of infection in graphs as in [1], [2] could also be inaccurate. Figure 1 illustrates this issue. Therefore, in this paper, we are focusing on the effect of graph connectivity and infection flow to control the infection spreading process. Thus, we can determine which set of vertices that we should protect.

In social media such as Facebook, the relationships of users are often directed and have a particular amount of magnitude to influence their related counterparts [8]. In terms of spreading rumors or hoaxes, a follower of a user could get *infected* if the user he/she followed is posting fake news. Suppose that node d is followed by node b. This connection is represented as edge $b - d$. The magnitude of the connection also plays an important role, while the probability of a particular user post to appear in their follower timeline is not same. It depends on their interaction activities such as the number of comments

Table I: Terms and Symbols

Symbol	Definition and Description
$G(V, E)$	graph G with the vertex set V and the edge set E
A	adjacency matrix of graph G
$A(i, j)$	the ith row and jth column element of A
$A(i, :)$	the ith row element of adjacency matrix A
$A(:, j)$	the jth column element of adjacency matrix A
n	number of vertices in the graph
m	number of edges in the graph
λ	largest eigenvalue of adjacency matrix A
μ	corresponding eigenvector of λ
$deg^+(i, j)$	outdegree of edge connecting vertex i and j
$w(i, j)$	weight of edge connecting vertex i and j
$outd(i)$	weighted outdegree of vertex i
$PS(i)$	Protection Score of node i
β	infection rate
δ	recovery rate
ϕ	number of initial infected nodes in a graph
θ	survival nodes percentage of graph

and likes [8]. As an illustration, the magnitude of edge $b - d$ is lower than edge $g - d$ for the reason that user b give more comments and likes to user d status which then increases their interaction activities. In this case, if user d post a rumor or fake news, user b have a higher probability to be *infected* than user g. In this case, being *infected* means that user b also posts or shares the rumor or fake news from user d.

The main contributions of our paper can be summarized as the following three points:

1. **Problem Formulation:** We formulate the Flow-Aware Vertex Protection problem, which considers the magnitude and infection flow of the large scale graphs arising from social media and epidemiology. This problem elaborates the directionality and weight of connection and their effects to infection flow of failure networks, which has been ignored by previous recent studies as in [1]–[4]. We introduce that the problem is NP-Hard and show its definition which have not been comprehensively studied before.

2. **Efficient Algorithm:** We develop an efficient, scalable algorithms for the problem, called GraphShield. Due to the complexity of FAVP problem that is hard to approximate within absolute error, we find that GraphShield is scalable for large graphs and gives effective protection compare with other popular methods. The algorithm utilizes the concept of graph connectivity from the largest eigenvalue and considering the flow of infection and outdegree centrality, then allocate selective priority on vertices regarding their weighted connection.

3. **Extensive Evaluations:** We perform comprehensive experimental simulation on multiple real network datasets to demonstrate its effectiveness and efficiency. Our proposed algorithms outperform several other existing methods, such as NetShield, NetShield+, Degree, PageRank, SubGraph, etc.

II. Related Works

Influence Maximization. The vertex protection problem and the influence maximization share the similar goal to find a set of vertices to control the influence propagate in the network. Kempe et al. have widely known for their work which firstly establishes the study in this field [9]–[11]. However, they are different with each other regarding the process [1]. The influence maximization select set of vertices

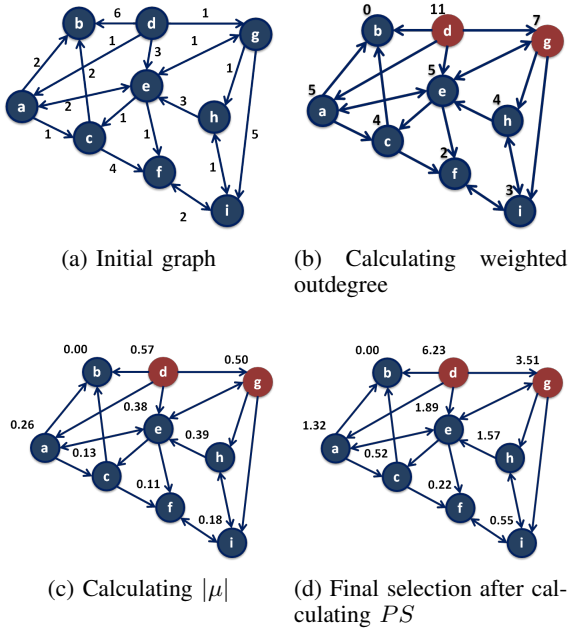


Figure 2: **GraphShield Algorithm.** Example of $k = 2$

to maximize the infected population [9], [11]. The vertex protection tries to minimize the infection spreading. Thus, our work is also different with theirs as a reaction to prevent the influence/infection spreading by removing connection of a certain set of nodes to protect the whole networks. These two processes could also happen at the same time in a graph.

Importance Measurement of Vertices on Graphs. Various centrality measurements take an important role in measuring vertex importance on the graph including the closeness centrality [7], PageRank [6] and the degree centrality [5]. Most of these methods quantify the score for each individual vertex, while our work focusing on collective quantification of k vertices regarding their effect in preventing infection spreading. Moreover, the centrality measurements are not specifically designed for protection or immunization objective [5]–[7]. Many of them are also not scalable enough to handling the large graphs.

To summarize, none of the related works are focused on the study of flow-aware vertex protection problem and approach in preventing spreading failures in directed large graphs.

III. Flow-Aware Vertex Protection Problem

Consider the input graph in an epidemic network as a directed graph $G = (V, E)$, vertices can have varied states (such as Susceptible, Infected, Recovered, etc.) based on the epidemic model. We consider SIS model in this work [12]. Here we first specify a formal definition of vertex protection problem as a background motivation for this work; then we formalize the definition of FAVP problem. Table I provides the main terms and symbols used in this paper.

Definition 1. Vertex Protection Problem

Let $G = (V, E)$ be a large un-directed un-weighted connected graph as an input with n vertices, SIS propagation model and a budget k. We define θ to be the percentage of vertices that remain uninfected at the end of infection propagation. Our goal is to find S, a subset of k vertices such

that θ is maximized. We get a new graph $G^{(S)}$ with adjacency matrix $\hat{A}$ by removing coresponding edges of S from G.

Definition 2. FAVP Problem

Given $G = (V, E)$, a large directed weighted connected graph with adjacency matrix A, budget k, SIS propagation model with propagation probability β and curing probability δ. We consider λ and μ to be the largest eigenvalue of A and its corresponding eigenvector as the connectivity measurement of nodes in the graph. We also consider deg^+ and w as outdegree and weight of edges as the possibility of infection flow. We define PS as the necessity of certain subset of vertices to be protected considering the connectivity and flow of infection. Our goal is to find S, a subset of k vertices with the highest PS among all possible subsets so that θ is maximized.

Theorem 1. FAVP Problem is NP-Hard.

Proof. Zhang et al. [2] has presented that Data-Aware Vaccination (DAV) problem is NP-Hard by reducing Minimum K-Union (MinKU) set problem [13] which was proven to be hard. We can reduce the MinKU problem to an instance of FAVP problem with $\delta = 1$ and $\beta = 1$, given that MinKU has instance a set S where $S_i \subseteq V$ and positif integer k. Hence, the FAVP problem under SIS propagation model for any given δ and β is also NP-Hard. □

IV. GraphShield Method

A. Intuition

Many literatures state that the largest eigenvalue λ of a graph G also represent the capacity of the graph in terms of loop capacity and path capacity [1], [14]. The larger λ of the graph, the better graph connected [1]. Van Dam et al. [15] and Wang et al. [14] also show that the smaller the largest eigenvalue λ, the larger the robustness of a network during infection propagation.

In infection spreading modeling, there is a critical state that infection spreading becomes endemic, which is called epidemic threshold [2]. Wang et al. demonstrated the prediction of this state using the largest eigenvalue since the threshold relies upon the structure of the graph [14].

B. GraphShield Algorithm

To protect the best set of vertices during infection spreading, we consider a metric for each node representing its necessity to be protected. The higher score of the metric, the higher importance of the node to be selected in the protection scheme.

We consider criteria to develop this metric, namely the Protection Score (PS), which can be summarized as follows:

The connectivity and vulnerability. As we already provided the main characteristic of the largest eigenvalue (λ) of a graph, we set its corresponding eigenvectors (μ) as variables.

The outdegree centrality. The role of degree centrality in networks has been discussed in many studies [5], [16]. There are many advantage of prioritizing the high degree vertices among the other [5]. In this work, we incorporate the benefit of this centralization to our method and modify it to be inline with infection spreading process. Thus, we consider the weighted outdegree ($outd$), the number of ties that the vertex are able to *infect* their neighbors with respect to the weight of edges among them. Regard to undirected graph, this outdegree is simply turns into degree centrality, counting the number of neighbors that a certain vertex can infects.

Algorithm 1: GraphShield

Data: Graph $G = (V, E)$
Input: the adjacency matrix A and an integer k
Output: a set S of k vertices

1 Let D be the degree matrix of A;
2 Compute the largest eigenvalue λ of A;
3 Let μ be the corresponding eigenvector of λ where $\mu(i)$ $(i = 1, ..., n)$;
4 Compute the outdegree of A;
5 Let $outd(i)$ be the vector element of weighted outdegree for $i = 1, ..., n$;
6 Initialize S to be empty;
7 **begin**
8 **for** $i \leftarrow 1$ **to** n **do**
9 **if** $outd(i) <> 0$ **then**
10 $PS(i) \leftarrow |\mu(i)outd(i)|$;
11 **else**
12 $PS(i) \leftarrow 0$;
13 **end**
14 **for** $iter = 1$ **to** k **do**
15 Let $j \leftarrow argmax_i PS(i)$, add j to set S;
16 **end**
17 **return** S
18 **end**

The flow of infection. Outdegree metric (deg^+) represents the possibility of certain infected nodes to infect their neighbors. In the case of infection spreading, this metric is highly associated with how to control and prevent cascading infection. We use the weighted outdegree ($outd$) in this work. In undirected case, we can consider the edges as having indegree and outdegree as well.

The Protection Score (PS) can be formulized as:

$$PS(i) = \begin{cases} 0, \text{if } outd(i) = 0 \\ \sum_{i \in V} |\mu(i)outd(i)|, \text{otherwise} \end{cases} \quad (1)$$

Algorithm 1 provides the detail of our proposed GraphShield. It provides a set S of k vertices as the output and requires the adjacency matrix A and an integer k as the input. We compute the largest eigenvalue (λ) and its corresponding eigenvectors (μ) from matrix A. We assign PS value of each vertice in step 8-13. Note that in step 11-12 we assign zero value if the outdegree is equal to zero. Figure 2 gives the example of selecting a set of vertices from a certain graph.

Here we will also compare our proposed GraphShield with NetShield by Chen et al. [1]. NetShield utilizes a metric, called Shield-value (Sv) which defined as:

$$Sv(S) = \sum_{i \in S} 2\lambda\mu(i)^2 - \sum_{i,j \in S} A(i,j)\mu(i)\mu(j) \quad (2)$$

There are some main critical differences of our approach and NetShield [1]. *First*, NetShield ignores the role of weight and direction in the development of protection metric, called

Shield-Value. *Second*, NetShield ignores the importance of degree centrality. We assume that both of graph connectivity (represented by the largest eigenvalue) and degree centrality plays an essential role in controlling the failure propagation. Hence, we combine the advantages of degree centrality and connectivity to determine the set of k vertices. *Third*, NetShield tends to select the set of k vertices which diverse among themselves by avoiding the adjacent vertices to be selected together. Notice that the set of highest degree vertices is not always the highest μ, vice versa [1]. In addition, Newman demonstrates the existence of degree-correlation among adjacent vertices in real-world networks [17]. Thus, we assume that as long as the adjacent vertices have high outdegree centrality, which also means they have the probability of *infecting* lots of their neighbors, those adjacent vertices can also be selected.

Reproducibility. For the repeatability of simulation results, we make the proposed GraphShield code available online[1].

Here we will provide the analysis of GraphShield algorithm in terms of computational complexity and cost of space. We analyze the efficiency based on Chen et al. [1] as a benchmark.

Computational Complexity. The cost of step 2 in Algorithm 1 is $O(m)$ using the power method, while we know that the cost of step 1,3,5, and 6 are constant. Steps 8-13 cost $O(n)$. For steps 14-16, its cost is $O(k)$.

$$\begin{aligned} cost(GraphShield) &= O(m) + O(n) + O(k) \\ &= O(n + m + k) \end{aligned} \tag{3}$$

Cost of Space. The space cost of steps 1-5 in Algorithm 1 are $O(n + m + 1) : O(m)$ for storing the graph, $O(m)$ for storing the degree matrix, $O(n + m)$ for running the eigen-decomposition algorithm, $O(1)$ for storing λ, $O(n)$ for storing μ, and $O(n)$ for storing the weighted outdegree ($outd$). The cost for step 6 is $O(1)$. The space cost of steps 8-13 is $O(n)$ which re-usable during the iteration. Lastly, to store the output S set of nodes, we need $O(k)$. By ignoring the constant term, we can summarize that the space cost of Algorithm 1 is

$$space(GraphShield) = O(n + m + k) \tag{4}$$

NetShield and NetShield+ as the state-of-the-art algorithm have computational complexity $O(nk^2 + m)$ and $O(mk/b + nkb)$ respectively [1]. b in NetShield+ is a batch size. Instead of selecting all the k vertices in one round, NetShield+ picks b vertices for current graph at each iteration, and then use the updated graph for next iteration until all k vertices are selected. The NetShield and NetShield+ have the same cost of space, $O(n + m + k)$. Thus, our proposed method has less computational complexity and require the same cost of space. Hence it is faster and more efficient in terms of cost of space.

V. Evaluations

A. Experimental Settings

Datasets. We run our experiments on various real network datasets. Our approaches are naturally able to generalize in undirected case by assuming the bidirectionality of the edges.

[1]http://bit.ly/GraphShield

Table II: Dataset

Name	#nodes	#edges
Undirected Unweighted		
Karate	34	152
Contiguous	49	107
Dolphins	62	159
EuroRoad	1,174	1,417
Facebook Ego	2,8888	2,981
Oregon	13,947	30,584
DBLP Coauthorship	317,080	1,049,866
Directed Unweighted		
Moreno-Hens	32	496
DBLP Citation	12,591	49,743
Cora Citation	23,166	91,500
GooglePlus Ego	23,628	39,242
Twitter Lists	23,370	33,101
Digg Friends	279,630	1,731,653
Citeseer Citation	384,413	1,751,463
Yahoo Advertisers	653,260	2,931,708
Directed Weighted		
Moreno-Rhesus	16	111
Moreno-Cattles	28	217
Moreno-Sheep	28	250
Moreno-Highschool	70	316
NeuralNetwork	297	2,345
US Airports	1,574	28,236

Therefore three types of cases are provided: undirected unweighted, directed unweighted and directed weighted graphs as summarized in Table II. All the dataset could be accessed at [18], except the Neural Network [19] and the Oregon [1].

Parameters. For effectiveness evaluation, we use the same parameters for all methods: $\beta = 0.9$ and $\delta = 0.6$, as used in [2]; $k = 20\%$ for smaller datasets (less than 200 vertices), $k = 10\%$ for larger datasets, and $\phi = k$ as suggested by [1]. We perform random initialization to determine the infected nodes in each simulation. On each dataset, we conduct 100 times of simulations and take the average. As for efficiency evaluations, various budget k value are used. For NetShield+, we use the batch size = 2.

Evaluation Metric. We compare the protection effectiveness result of all methods using a survival nodes percentage (θ). To measure the efficiency, we compare computational time for various value of the budget k on large graphs.

Comparison Methods. We compare our proposed method with popular methods: Degree centrality [5], PageRank [6], EigenVector [5], SubGraph [5], KCoreness [5], Closeness [7], Greedy [1], and also including the state-of-the-art methods: NetShield and NetShield+ [1].

B. Effectiveness Evaluation

We simulate the methods on random attack and provide average after 100 simulations. Table III shows that our GraphShield outperforms the other methods regarding the highest average of survival nodes percentage (θ) at the end of propagation after 100 timesteps. GraphShield also performs well in undirected graphs by assuming the bi-directionality of edges and threat the outdegree parameter in the PS score as a degree centrality. Hence, it combines the role of degree centrality and largest eigenvalue. In the directed graphs (Moreno-Hens) and directed weighted graphs (Neural Network, US

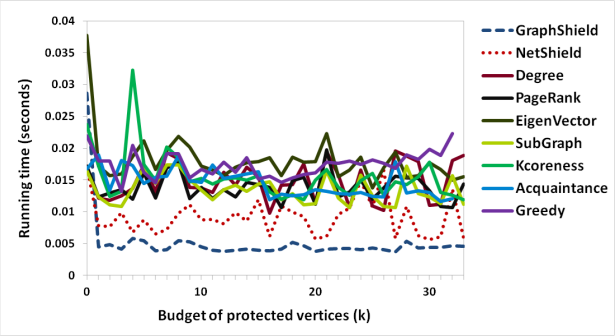


Figure 3: **Efficiency Evaluation.** Running time comparison on Karate dataset.

Airport, and Moreno-Highschool), the GraphShield are able to tackle the infection flow limitation of the current state-of-the-art methods: NetShield and NetShield+.

C. *Efficiency Evaluation*

Here we evaluate the running time of the proposed methods. All of the experiments are done on the same machine with Intel Core i7-3770 CPU @ 3.40GHz and 8 GB memory, running Ubuntu 16.04. Figure 3 shows that for different budget k, GraphShield takes less running time than the other methods. Notice that NetShield+ is not displayed here due to its long running time, which will be evaluated in Figure 4. In addition, the NetShield is faster than other methods, except GraphShield, which reconfirm its efficiency as in [1].

Next, we will show the performance of GraphShield against the current state-of-the-art methods, NetShield and NetShield+ [1] on large graphs. Here we only provide the evaluation on larger graphs. Figure 4 shows the running time superiority of GraphShield over the NetShield+. Moreover, from each left charts in Figure 4, the proposed GraphShield is also faster than the NetShield. NetShield+ performs not so good on large graphs due to its batching processes.

VI. Discussions

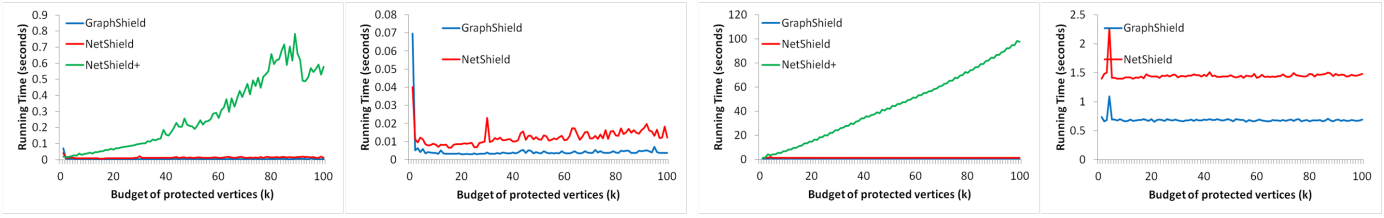
As the connection of large social networks is not merely binary entities, flow-aware protection problem representation is required. The limitation of our approach might appear from ignoring the role of in.degree, which may essentially take part in the imploding case of the graph structure. The case of undirected graphs can be handled by assuming the bi-directionality of edges. On the other hand, the existence of self-loop is ignored in our work due to its limited effect on neighborhood infection process. In the case of there exist the multiple edges among vertices, we treat them as a weight of edges and combine them into a single edge. However, this treatment can only be done on multiple edges in directed unweighted graphs. Thus, the further appropriate treatment for multiple edges in directed weighted graphs should be more evaluated. Regard to various targetted attack on graphs, instead of random attack as evaluated in this paper, is still not yet elaborated in our work, as well as in many current studies. All of these issues can be a good direction in the future.

VII. Conclusions

In this paper, we have addressed the problem of protecting set of vertices to prevent failure propagation process. We consider the role of infection flow and graph structures to formulate the Flow Aware Vertex Protection Problem. We proposed an efficient and scalable algorithm, called GraphShield to determine set of nodes we should protect given the limited k budget. Evaluation on various real graph datasets shows that the GraphShield outperforms the state-of-the-art algorithms in

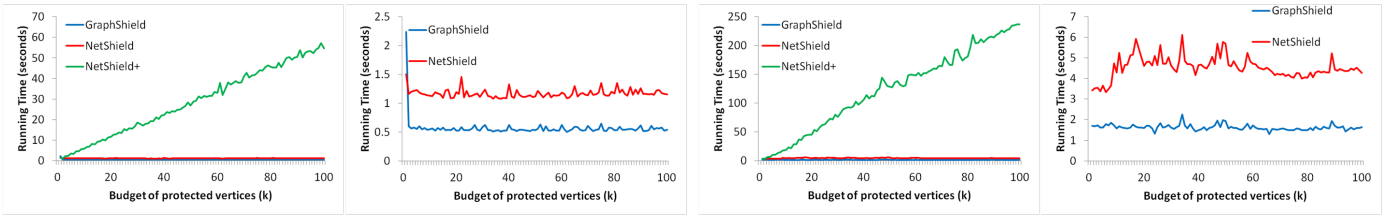
Table III: Effectiveness Evaluation

	Facebook Ego		Neural Network	
Method	Average	Std.Dev.	Average	Std.Dev.
Degree	88.17	8.43	70.79	0.32
PageRank	89.78	8.06	70.72	0.31
EigenVector	89.44	7.77	70.74	0.37
SubGraph	89.03	7.43	70.71	0.38
Kcoreness	88.04	7.91	70.78	0.34
Greedy	89.66	8.11	70.82	0.32
NetShield	88.58	7.29	70.72	0.32
NetShield+	88.86	7.77	70.74	0.31
GraphShield	**90.59**	7.79	**70.86**	0.34
	Dolphins		US Airports	
Method	Average	Std.Dev.	Average	Std.Dev.
Degree	85.92	2.00	73.19	0.13
PageRank	86.39	2.43	73.19	0.12
EigenVector	86.16	2.16	73.19	0.14
SubGraph	86.18	2.44	73.22	0.14
Kcoreness	86.34	2.41	73.21	0.13
Greedy	86.11	2.29	73.21	0.15
NetShield	85.81	2.05	73.19	0.13
NetShield+	86.03	2.54	73.15	0.15
GraphShield	**86.85**	2.39	**73.26**	0.15
	Contiguous		EuroRoad	
Method	Average	Std.Dev.	Average	Std.Dev.
Degree	88.14	3.57	69.24	0.48
PageRank	87.69	3.17	69.40	0.53
EigenVector	88.37	3.35	69.29	0.55
SubGraph	88.31	3.28	69.33	0.49
Kcoreness	87.98	3.86	69.33	0.51
Greedy	87.92	3.69	69.30	0.50
NetShield	87.69	3.52	69.28	0.52
NetShield+	88.02	3.52	69.32	0.49
GraphShield	**89.14**	3.95	**69.51**	0.55
	Moreno-Hens		Moreno-Highschool	
Method	Average	Std.Dev.	Average	Std.Dev.
Degree	70.06	1.55	70.40	0.81
PageRank	69.63	1.41	70.49	0.89
EigenVector	69.75	1.47	70.37	0.83
SubGraph	69.75	1.47	70.30	0.74
Kcoreness	69.91	1.52	70.49	0.82
Greedy	70.09	1.55	70.43	0.80
NetShield	69.97	1.53	70.37	0.83
NetShield+	69.78	1.48	70.54	0.83
GraphShield	**70.25**	1.57	**70.64**	0.87
	Moreno-Sheep		Karate	
Method	Average	Std.Dev.	Average	Std.Dev
Degree	72.93	2.44	77.47	1.40
PageRank	73.21	2.61	77.38	1.37
EigenVector	73.32	2.56	77.85	2.66
SubGraph	73.18	2.56	77.56	1.43
Kcoreness	73.18	2.51	77.79	1.47
Greedy	73.25	2.61	77.79	1.47
NetShield	73.25	2.61	77.68	1.45
NetShield+	73.32	2.61	77.62	1.44
GraphShield	**74.79**	3.00	**78.06**	1.47



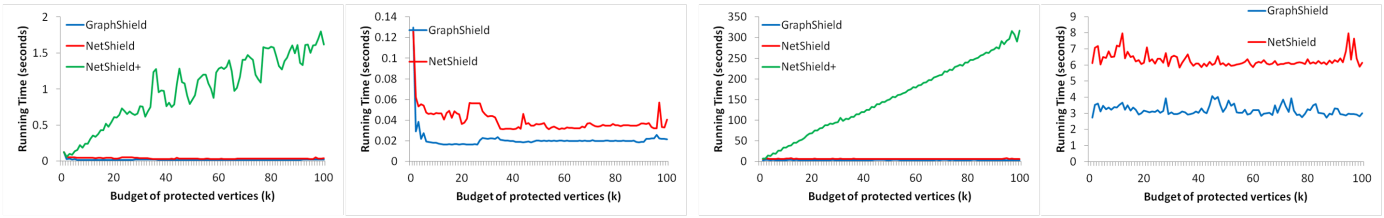
(a) Facebook Ego

(b) DBLP Coauthorship



(c) Digg Friends

(d) Citeseer Citation



(e) GooglePlus Ego

(f) Yahoo Advertisers

Figure 4: **Running Time Evaluation.** Running time (x-axis) versus changing of the budget k (y-axis). In each subfigure, the left chart shows the running time comparison of GraphShield, NetShield, and NetShield+. The right chart shows only the comparison of GraphShield and NetShield for a better visualization. GraphShield shows it scalability on large graphs and performs faster than the current state-of-the-art methods: NetShield and NetShield+. Lower is better.

terms of both effectiveness and efficiency.

ACKNOWLEDGEMENT

This work was supported by Tokyo Tech - Fuji Xerox Cooperative Research (Project Code KY260195), JSPS Grant-in-Aid for Scientific Research(B) (Grant Number 17H01785) and JST CREST (Grant Number JPMJCR1687). A.W.W. also thanks to Indonesia Endowment Fund for Education (LPDP) for the educational scholarship.

REFERENCES

[1] C. Chen, H. Tong, B. A. Prakash, C. E. Tsourakakis, T. Eliassi-Rad, C. Faloutsos, and D. H. Chau, "Node immunization on large graphs: Theory and algorithms," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 1, pp. 113–126, Jan 2016.

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

[3] Y. Zhang and B. A. Prakash, "Data-aware vaccine allocation over large networks," *ACM Trans. Knowl. Discov. Data*, vol. 10, no. 2, pp. 20:1–20:32, Oct. 2015.

[4] 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*, ser. CIKM '14. New York, NY, USA: ACM, 2014, pp. 1719–1728.

[5] M. Newman, *Networks: An Introduction*. New York, NY, USA: Oxford University Press, Inc., 2010.

[6] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web." Technical Report 1999-66, November 1999, previous number = SIDL-WP-1999-0120.

[7] C. Dangalchev, "Residual closeness in networks," *Physica A: Statistical Mechanics and its Applications*, vol. 365, no. 2, pp. 556–564, 2006.

[8] Facebook. (2017) How does news feed decide which stories to show? [Online]. Available: https://www.facebook.com/help/166738576721085

[9] D. Kempe, J. Kleinberg, and E. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '03. New York, NY, USA: ACM, 2003, pp. 137–146.

[10] K. Jung, W. Heo, and W. Chen, "Irie: Scalable and robust influence maximization in social networks," in *2012 IEEE 12th International Conference on Data Mining*, Dec 2012, pp. 918–923.

[11] P. Shakarian, A. Bhatnagar, A. Aleali, E. Shaabani, and R. Guo, *The Independent Cascade and Linear Threshold Models*. Cham: Springer International Publishing, 2015, pp. 35–48.

[12] A. Gray, D. Greenhalgh, L. Hu, X. Mao, and J. Pan, "A stochastic differential equation sis epidemic model," *SIAM Journal on Applied Mathematics*, vol. 71, no. 3, pp. 876–902, 2011.

[13] S. A. Vinterbo, "Privacy: a machine learning view," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 8, pp. 939–948, Aug 2004.

[14] Y. Wang, D. Chakrabarti, C. Wang, and C. Faloutsos, "Epidemic spreading in real networks: an eigenvalue viewpoint," in *22nd International Symposium on Reliable Distributed Systems, 2003. Proceedings.*, Oct 2003, pp. 25–34.

[15] E. van Dam and R. Kooij, "The minimal spectral radius of graphs with a given diameter," *Linear Algebra and its Applications*, vol. 423, no. 2, pp. 408 – 419, 2007.

[16] S. P. Borgatti, "Centrality and network flow," *Social Networks*, vol. 27, no. 1, pp. 55 – 71, 2005.

[17] M. E. J. Newman, "Assortative mixing in networks," *Phys. Rev. Lett.*, vol. 89, p. 208701, Oct 2002.

[18] J. Kunegis, "Konect - the koblenz network collection," in *Proc. Int. Conf. on World Wide Web Companion*, 2013, pp. 1343–1350.

[19] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, pp. 440–442, 1998.