sensors

MDPI

Article

Node Immunization with Time-Sensitive Restrictions

Wen Cui [1], Xiaoqing Gong [1], Chen Liu [1], Dan Xu [1], Xiaojiang Chen [1], Dingyi Fang [1,*], Shaojie Tang [2,*], Fan Wu [3] and Guihai Chen [3]

[1] School of Information and Technology, Northwest University, Xi'an 710127, China; cuinwu@gmail.com (W.C.); gxq@nwu.edu.cn (X.G.); liuchen@nwu.edu.cn (C.L.); xudan@nwu.edu.cn (D.X.); xjchen@nwu.edu.cn (X.C.)

[2] Naveen Jindal School of Management, University of Texas at Dallas, Richardson, TX 75080, USA

[3] Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China; fwu@cs.sjtu.edu.cn (F.W.); gchen@cs.sjtu.edu.cn (G.C.)

* Correspondence: dyf@nwu.edu.cn (D.F.); tangshaojie@gmail.com (S.T.)

Academic Editors: Mianxiong Dong, Zhi Liu, Anfeng Liu and Didier El Baz
Received: 9 October 2016; Accepted: 5 December 2016; Published: 15 December 2016

Abstract: When we encounter a malicious rumor or an infectious disease outbreak, immunizing k nodes of the relevant network with limited resources is always treated as an extremely effective method. The key challenge is how we can insulate limited nodes to minimize the propagation of those contagious things. In previous works, the best k immunised nodes are selected by learning the initial status of nodes and their strategies even if there is no feedback in the propagation process, which eventually leads to ineffective performance of their solutions. In this paper, we design a novel vaccines placement strategy for protecting much more healthy nodes from being infected by infectious nodes. The main idea of our solution is that we are not only utilizing the status of changing nodes as auxiliary knowledge to adjust our scheme, but also comparing the performance of vaccines in various transmission slots. Thus, our solution has a better chance to get more benefit from these limited vaccines. Extensive experiments have been conducted on several real-world data sets and the results have shown that our algorithm has a better performance than previous works.

Keywords: node immunization; strategy; social network

1. Introduction

To what extent have you been bothered by the outbreak of undesirable things, a contagious disease, computer virus, or malicious rumor, and gone through a difficult time to wait for the constraint of the outbreak? Minimizing the spread of undesirable things is a major problem in Centers for Disease Control, Kaspersky, Twitter and Facebook, etc., where management and staff undertake much more responsibility for discovering and controlling the contamination. Many organizations are looking to address this problem with the distribution of vaccines/patches in the relevant network with a limited budget [1–3]. For example, before the outbreak of H1N1 Flu, American Centers for Disease Control and Prevention(CDC) had centralized the distribution of vaccines for nearly 150,000 sites (hospitals, clinics, health departments), and it had been based on a cautious strategy of distributing the limited budget of vaccines to protect more people from being infected. Current placement strategies of vaccines/patches, however, are assuming that the initially infected nodes are randomly appearing [4–7]. Almost all cases, however, are such that the initially infected nodes are unevenly distributed. For example, in a city, some susceptible individuals are at high risk of getting an infection from a particular infectious disease (e.g., poultry farmers who are easily exposed to bird flu). Similarly, once a computer virus was made, it is aiming at the servers which lack specific precautions rather than randomly attack servers, and the propagation of this virus will start from those servers, obviously. Knowing the initially infected nodes as a prior information is also indispensable for the rumor control of social

network companies. They could send an official alert message to limited people to prevent the rumor from developing into a public affair. With the help of a reliable node protection strategy, the related social network applications also can be extended to other hot topics, such as minimizing the error message propagation in Cyber-Physical Systems [8–14], which already attracts a lot of attention from the academic field and industry areas [15–19]. Besides, the dissemination of data in a network [20–24] can also be guaranteed in a better way.

Minimizing the influence of undesirable things under prior information has been recently proposed [25,26] and the key part of solving this problem is to propose a greedy vaccines placement algorithm. Researchers try to improve protection performance of their algorithm by modifying heuristic idea since the problem is **NP-hard** [27] and even computing the influence between each node is a **#p** problem [28]. However, underlying their solutions is an assumption that all vaccines need to be deployed at once, which does not relate well to the facts. The saving number of one vaccine may have a notable difference at a different time with various network structures. Thus, the immunization strategy should take the benefit of each vaccine at different time slot into consideration so that we can maximize the utilization of every vaccine. This paper introduces Dancing with the propagation virus (DWPV), a dynamic vaccines placement strategy combines the time division into our solution. In line with common practice in selecting vaccines set [25,26,29,30], DWPV adopts Independent Cascade (IC) model [27] as virus propagation model. To get the placement position of vaccines, the propagation model is a basic function to analyze the influence of immunizing those candidate nodes. The challenge of our problem, however, is to identify where and when the vaccines should be distributed.

Unlike past approaches [6,7], which consider delaying the placement time as detrimental, DWPV exploits the postponement of vaccine distribution to protect healthy nodes from being infected. Specifically, the benefit of putting one vaccine in a specific position on the network is always changed by the dynamic network structure. Hence, if we can ensure that a higher benefit of vaccines can be obtained from other transmission time slot (as shown in Figure 1), then we postpone the placement time of those vaccines to next time slot to save more nodes.

To illustrate DWPV's approach, Figure 2 shows a toy example with some susceptible nodes and one infected node, where the infected node is surrounded by three neighbors A, B and C, and the virus propagation probability is 0.2. As the figure shows, the idle vaccine, like the yellow one, may have a very limited benefit from the neighbors of the infected node at time slot 1 as we are not really sure who will be infected in the next time slot. However, this vaccine tends to have significantly different benefit from being distributed at time slot 2, e.g., putting the vaccine on A'. Hence, one needs to consider the full-time series benefits of vaccines; one-time slot alone, like the benefit from time slot 1, can be invalidated. To see this more clearly, we can use the standard metric [25] to measure the effectiveness of putting vaccine on the each node of Figure 2. Figure 1 shows the obtained benefit from each node of Figure 2 at t = 0 and t = 1. It is clear from this figure that looking at the neighbors of initially infected alone—i.e., A, B and C at t = 0—would wrongly indicate that vaccinating node B is the best choice, while looking at t = 1 allows us to realize that vaccinating node A' at t = 1 has more benefit than vaccinating node B at t = 0. Thus, a robust vaccine placement strategy needs to compare the node's benefit at different time slots by quantifying how much difference there is between putting vaccines on the current time slot and next time slot.

So how can we automatically quantify the changes across the benefit of different time slots? To do so, we need to estimate the influence of nodes' benefit in different time slots. In contrast to the illustrative example, however, real-world networks always have many potential propagation events, which may change differently depending on the number of initially infected nodes and the propagation probability of each node. Further, different scale of networks typically has a different complexity of relationship between each node, causing the benefit of nodes to be changed frequently. In designing a technique that finds the most suitable node and its placement time despite these changes, we are inspired by the baseline strategy [25], which without consideration the placement time of vaccines, in using the dominator tree to approximately represent the relationship between each node of the

network. Given an arbitrary graph and trying to figure out the relation between each node is a **#p** problem to us, we need to approximately represent the dominating relation to finding the dominator node that we can put a vaccine on there to release the most infecting pressure of other nodes. In graph theory, a graph is converted to a dominator tree following the rules that if the status of A is donated by B then we define it as B denote A. Namely, we can directly point out the dominating node after we translate the original graph into a tree structure and merge the initially infected nodes as the root node. As opposed to only converting the original graph into the dominator tree and point out the key node, DWPV analyzes the transmission probability of each root node's neighbor, which nodes are treated as the best candidate nodes in previous works, by using a binary tree to figure out the expected benefit of these nodes in current time slot and next time slot. Furthermore, we propose a Monte Carlo sampling method as a substitute method to suit the limited computation application. We present the design in Section 4 and we have theoretically proved that the strategy of ours, which combine the time division into the placement of vaccine, has an advantage over previous works in controlling the propagation of the virus.

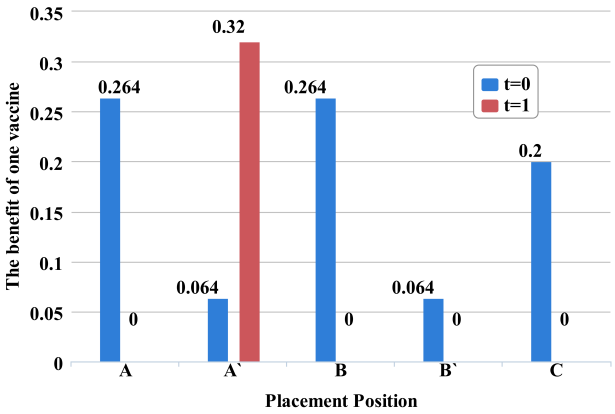


Figure 1. Different benefit of position over each placement time.

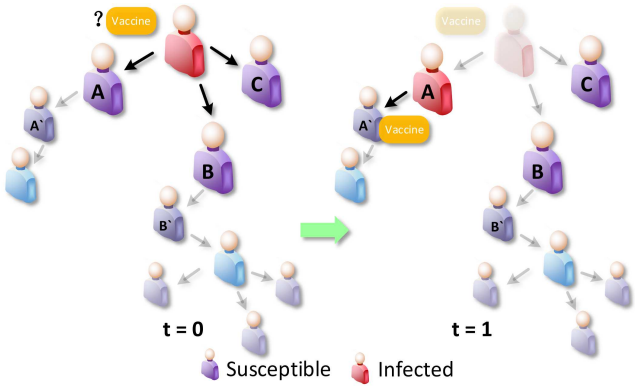


Figure 2. Intuition underlying DWPV use of time division in distributing vaccines: The figure shows an infected node (in red) and some susceptible nodes (in blue). The infected node has one chance to infect their neighbors at $t = 0$. At $t = 1$, the node A has been infected by the initially infected node and other nodes (B and C) are lucky ones.

Finally, while we could obtain a set of position to put the vaccines and its corresponding placement time, there will always be many changes in the propagation process of the virus, since all analysis results are based on a probabilistic model. For this reason, DWPV decides to put one vaccine on the graph only if a neighbor of the candidate node is infected. This design broadens the assurance of

the effectiveness of vaccines (up to 30%), and in case of some vaccines, they become invalid as their neighbors are always safe during the propagation of the virus.

This paper makes the following contributions:

1. It presents a vaccine placement strategy that exploits the changes of vaccine's benefit in different time slots to get the highest benefit of one vaccine. As a result, the design delivers high-level protection to keep more healthy nodes from being infected.
2. It also demonstrates the capability of the dominator tree to select out the candidate immunization nodes in a dynamic scenario, and we successfully use it to protect more healthy nodes in an infected network than the static one. While our design and results are presented in the social network, the basic idea can be extended to other communication problems.
3. It presents an accurate method to evaluate the possible benefit from other time slots by using binary tree and its simplified version with lower computation complex by using Monte Carlo method.
4. It has been evaluated in some real-world large-scale data sets.

The rest of the paper is organized as follows. Section 2 introduces our preliminaries. Section 3 define our problem as vaccines placement problem and we present our solution in Section 4. Section 5 shows our experiment results with open-resource datasets. Section 6 is the related work about this paper and we have a conclusion of this paper in Section 7.

2. Preliminaries

The information propagation model is the first thing we have to understand for knowing the method of restricting the malicious things communication in the social network, which is depicting the propagation process of information (bad/good) in a discrete way. The Susceptible/Infected/Recovered (SIR) Model [1] and Independent cascade (IC) model [27] are two popular models about malicious information communication which have been well studied for quite a long time. Compare to the SHIR (Susceptible-Hidden-Infected-Recovered), SIS(Susceptible-Infected-Susceptible) and LT (Linear Threshold), those two models are more widely used. In this paper, we introduce our method in IC model first, and we transfer the IC model into SIR model later. Meanwhile, notations list is shown in Table 1.

Table 1. List of notations.

Notation	Definition
$G(V, E)$	a propagation network, in this paper, we define our problem on undirected graph but not limited
$P_{u,v}$	the probability of u infected by v, and vice versa
SIR	transmission model with three status, *susceptible* (healthy), *infected* and *recovery*
R	recovery probability, one node from *infected* to *recovery* in SIR model
IC	a SIR-like model but that every nodes have only one chance to infect their neighbors
I_0	initial infectious nodes set
$\mathcal{V}$	set of vaccines, which is used to stop or release the propagation of malicious things in $G(V, E)$
k	the budget of vaccines
I_i	infectious nodes set at time i
$\varphi_{G,I_0}(\mathcal{V})$	expected healthy nodes as the virus propagation process is over, and increasing its numerical value is the main target of this paper
$\beta(n)$	expected value of saving healthy nodes by vaccinating n in $G(V, E)$.

SIR Model is a comprehensive description model of disease spreading, which has been well studied for many decades. As shown in Figure 3, initial infected nodes I_0 are already in the graph $G(V, E)$. Besides, the weight of each edge, $P(u, v)$ $(0 < P(u, v) < 1)$, is used to represent the *transmission probability* between u and v. Here, if one node gets infected at step t, then that node will try to infect its neighbors (*susceptible*) in $G(V, E)$ at step $t + 1$. The status of each node will be updated in every time slot and it is notable that the recovered nodes will be removed from $G(V, E)$.

It is not until there is no infectious node in the $G(V, E)$ that the transmission processing of virus is over. Besides, once a node gets infected, that node will try to transform its status from *infected* to *recovery* (something like a man has antibodies) and the probability of this transformation is determined by $\mathcal{R}$.

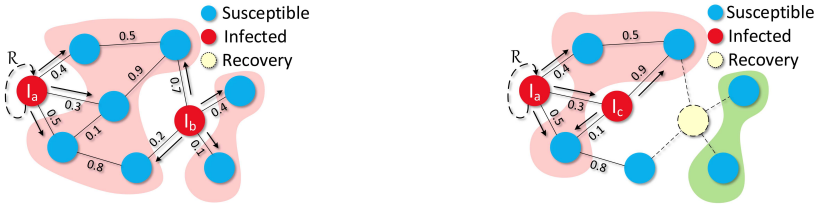


Figure 3. SIR transmission process: At $t = 0$, I_a and I_b are the initial infected nodes in $G(V, E)$. Those two nodes are trying to infect their neighbors through weighted connection edges. At the same time, both I_a and I_b are struggling to become recovered by using their own antibodies, the probability of occurrence of this event is R. At $t = 1$, a new infectious node I_c is coming and trying to infect its neighbors, fortunately, I_b overcame the hateful disease and protected its neighbors (which are in the green area) from being infected. (**a**) $t = 0$; (**b**) $t = 1$.

In general, in order to concentrate on the propagation process of the undesirable thing, we always use IC model to instead of SIR model [31–33]. Compared to SIR model, IC model omits the process of recovery ($\mathcal{R} = 1$) and every infected node has only one chance to infect its neighbors. In this paper, we adopt IC model as our basic model too, and we will introduce how to transform IC to SIR in Section 4.

3. Problem Definition

The influence minimization problem is like the *Vaccines Placement* (VP) problem, which is focusing on restricting the influence of malicious information in the social network. Based on the knowledge of information propagation model, we will give a formal definition of *Vaccines Placement* (VP) problem for knowing its optimization objective and problem complexity to specify a suitable method for our problem.

3.1. Vaccines Placement Problem

We are aiming at getting the maximum number of healthy nodes in the network, which means that we need the higher value of $\varphi_{G,I_0}(\mathcal{V})$ at $t = \mathcal{T}$ ($I_{\mathcal{T}} = \emptyset, t \in (0, \mathcal{T})$) in a weighted graph $G(V, E)$ with initial infectious nodes set I_0 and IC based propagation model. Namely, the target is to solve the *Vaccines Placement* (VP) problem that is to find the appropriate vaccines set $\mathcal{V}$ so that we can save the maximum number of nodes, where $\mathcal{V} \subset \mathcal{H} \subset V$ and $\mathcal{H}$ is the subset of V represent healthy nodes in $G(V, E)$. Formally,

$$\begin{aligned} &\max \ \varphi_{G,I_0}(\mathcal{V}) \\ &s.t. \quad |\mathcal{V}| = k. \end{aligned} \tag{1}$$

Once we have the formal description of our problem and the simple example of Figure 2, we could know that only knowing the position of vaccines is not enough for us to get the richest effectiveness of the limited vaccines. We should have taken the time division of immunization into our strategy to ensure the effectiveness. For this reason, we separate the original VP problem into two individual problems, a static one and dynamic one, to emphasize the importance of the immunization time and eventually get more healthy nodes in the network.

3.1.1. Static Vaccines Placement Problem

After being carefully selected from $\mathcal{H}$, naively, $\forall vaccine \subset \mathcal{V}$ needs to be put on $G(V, E)$ at $t = 0$, as if we cannot wait any longer. Then, we define this kind of problem which selects $\mathcal{V}$ only at $t = 0$ as SVP problem, since $\mathcal{V}$ is aiming at reducing the damage just caused by I_0 which has less consideration of the changes about $I_1 \to I_\mathcal{T}$.

3.1.2. Dynamic Vaccines Placement Problem

The main difference between DVP(G, I_i, t) and SVP(G, I_0) is that DVP(G, I_i, t) has a *time dimension* t. As the IC model is a stochastic model, we consider that the $\mathcal{V}$ may be not fixed. Specially, we take the deploying time of vaccines into consideration to save healthy nodes, where $i \subset (0, \mathcal{T})$, and we believe that $\mathcal{V}$ can be adjusted for $\max\{\varphi_{G,I_t}(\mathcal{V}) || \mathcal{V}| = k^*, \mathcal{V} \subset \mathcal{H}\}$ at different t, where k^* represents the changing value of k since some vaccines deployed at different t. Therefore, based on the above points of view, we can get the Remark 1 as follows.

Remark 1. *SVP problem is a special case of DVP problem, SVP* $\subset$ *DVP.*

3.2. Complexity of Vaccines Placement Problem

Unfortunately, the VP problem is **NP-hard** [27]. More over, the greedy algorithm cannot be easily used as the best approximated solution [25], since $\varphi_{G,I_0}(\mathcal{V})$ is not a submodular function, which means VP problem do not satisfied with the law of diminishing marginal utility that $f(\mathcal{V} \cup \{v\}) - f(\mathcal{V}) \not\geqslant f(\mathcal{U} \cup \{v\}) - f(\mathcal{U})$, where $\forall v \subset \mathcal{V}$ and $\mathcal{V} \subset \mathcal{U} \subset \mathcal{H}$.

Theorem 1. *Dynamic Vaccines Placement Problem is* ***NP-hard***

Proof. As mentioned in Remark 1 that SVP is a special case of DVP, and after adding the *time dimension* into SVP, the complexity of DVP is increased rather than be reduced. Namely, every selecting scheme has the same computation level as SVP has, and we need to perform the selecting algorithm multiple times at each time slot until k^* from k to 0. Furthermore, in [25], the SVP problem has been reduced from a MinKU set [34] problem which has been proved as an **NP-hard** problem. Thus, the DVP problem is not **NP-hard** either. □

From [28] we can ensure that the infection probability between two nodes cannot easily be acquired, which has been approved as **#p** problem [28]. Given any constant $0 < \epsilon < 1/3$, there exists a m_ϵ such that the SVP problem with $m > m_\epsilon$, cannot be approximated in polynomial time within an absolute error of $\frac{1}{2}m^{1-2\epsilon} + \frac{3}{8}m^{1-3\epsilon} - 1$ unless **P = NP**. Therefore, it becomes our prime purpose that we need to design a heuristic algorithm for solving the SVP problem.

4. Placement Strategy of Vaccines

Immunization time of limited vaccines, as we already knew, is an important key to satisfied our nodes protecting need. However, as the network structure is difficult enough that we can not get the immunization time and position easily from the original graph. Naturally, we need some basic tools of graph theory to initialize our network so that the structure can be simplified to meet the demand of analyzing the influence of each candidate node's position. Once we get the easy network, we could propose our heuristic idea of the VP problem and verify its advantages over previous static version.

4.1. Network Initialize

The VP problem has been proved to be an **NP-hard** problem in the previous section and it needs a heuristic method to be solved. As mentioned before, the essence of our problem is a graph problem. However, knowing the influence relationship between two nodes is a **#p** problem, and we need to design a method to solve our problem in polynomial time. Hence, we initialize the network to

simplify our problem. In this section, we will introduce the initialize methods, like dominator tree and *super infected nodes*.

4.1.1. Dominator Tree

For solving VP problem, after being given a graph $G(V, E)$ with initial infected nodes set I_0, the selection process of $\mathcal{V}$ not only need to find out the *Most Influence Node Set* (MINS) but also need to take the relationship between I_0 and MINS into consideration. Meanwhile, getting the relationship between each node is a **#p** problem, then we need an approximate method to represent the relationship. Luckily, the Dominator Tree (DT, as shown in Figure 4c that is a tree with infected root) has the power to convert a graph into tree structure with a dominating relation in linear-time, e.g., if node u wants to infect v, all the path between u and v need to pass m, then we define it as m dom v (or $m \rightarrow v$). After having a tree structure to represent $G(V, E)$, then we just need to treat those nodes, which are in the *Most Dominating Layer* (MDL) of DT (the first layer after the root node, e.g., the pink area as shown in Figure 4c, as the best candidates.

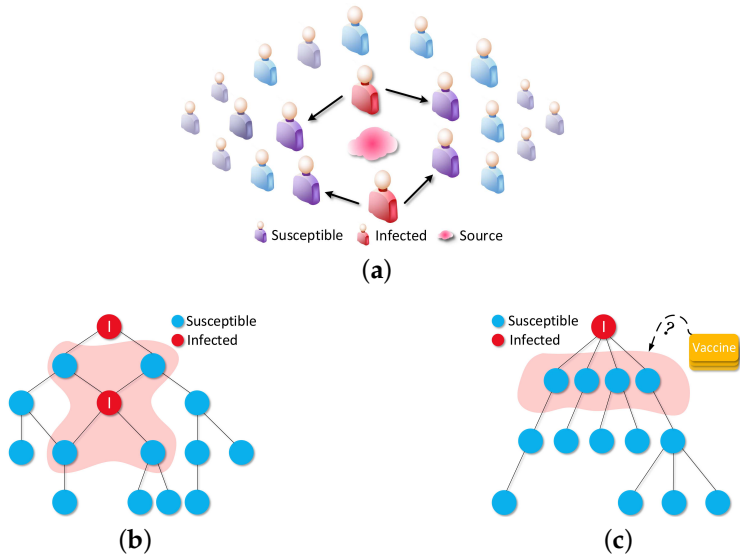


Figure 4. Fundamental Knowledge of Our Method: At first, some miserable people had caught a malignant virus and it would be spread to more and more people who have connection with those infected men; Then, we translated the abstract of social network into graph $G(V, E)$ for designing an appropriate method to control the malignant virus; At last, we convert $G(V, E)$ into DT by combing the infected nodes as one super infected node and let that super infected node to be the root of DT. Once we have DT, we just need to select out the *best-k-benefit* nodes in MDL. Finally, we can get the vaccines set $\mathcal{V}$ at $t = 0$ for the SVP problem. (**a**) People infected; (**b**) $G(V, E)$; (**c**) Dominating Tree(DT).

Definition 1 (Most Dominating Layer)**.** *After being given a DT with infected root, then we define the first layer after the root node as the Most Dominating Layer (MDL). MDL is a node set that nodes in this layer not only has the highest probability to be infected by the infectious root node of all other nodes but also can determine if their children will be infected or not. Obviously, $|MDL| < k$ is always true, otherwise we can easily stop the propagation of virus just by removing all direct suspectable nodes.*

4.1.2. Benefit of One Node

Calculating the influence of removing one node in $G(V, E)$ is a **#p** problem problem, which has been proved by [28]. However, in DT, we can calculate out the value of influence (we define it as *benefit*) with an easy method since each branch of a tree structure is mutually independent. Thus, we can treat

benefit as an estimation metric to estimate the value of candidate nodes so that we can make a better choice from those nodes.

Definition 2 (Metric for Estimating the Value of Removed Node)**.** *In DT, every node has a* benefit *which is used to represent the expected benefit of removing itself from $G(V, E)$, and that is defined as a metric for estimating the value of the removed node. As every branch in DT is independent, the calculation of the benefit is a linear time algorithm and the details has been shown in Algorithm 1.*

In Algorithm 1, $\beta(i)$ is calculated by a recursion function. Line 3–5 depict that the calculating process will be kept until it down to the leaf node. In DT, as for each branch is independent, the complexity of this algorithm is $\mathcal{O}(n)$. Meanwhile, a vivid instance of Algorithm 1 is shown in Figure 5.

Algorithm 1 Benefit of one candidate vaccine.

Input: One candidate vaccine i, $P_{u,v}$ and a dominator tree with infected root node I_0
Output: The benefit of node i and it is stated as $\beta(i)$
Function Cal-Benefit (node i)
1: **if** i is not a leaf node **then**
2: $\beta(i) = 1$
3: **for** each child j of node i **do**
4: $\beta(i) = \beta(i) + \textit{Cal-Benefit}(j) * P_{ij}$
5: **end for**
6: **else**
7: $\beta(i) = 1$
8: **end if**
9: **return** $\beta(i)$
EndFunction

Then, we can transform our target function (Equation (1)) into

$$\begin{aligned} &\max\ \phi_{DT,I_0}(\mathcal{V}) \\ &s.t.\quad |\mathcal{V}| = k. \end{aligned} \tag{2}$$

Here, we use the calculation of the maximum summation of expected benefit $\mathcal{V}_B$ to stands the selection of $\mathcal{V}$. Moreover, as $\mathcal{V}_B \propto \max\ \phi_{DT,I_0}(\mathcal{V})$, we can constrain our target into calculating the value of $\mathcal{V}_B$,

$$\mathcal{V}_B = \max \sum_{j=1}^{j=k} \beta_j(i), i \subseteq V. \tag{3}$$

4.2. The Heuristic Idea of VP Problem

From the point of adding time division into our solution, heuristically, we should compare the advantage of vaccines in each time slots and decide which slots is the best immunization time finally. Moreover, we discover that $\mathcal{V}_B$ can be obtained from other layers of DT by waiting for some time slots, rather than just from the MDL.

Lemma 1. *Given a DT with infected root node, we should not just treat the nodes in MDL as optimal candidates for Equation (2) since sometimes we can own a bigger $\mathcal{V}_B$ from other layers of DT by waiting some transmission time slots.*

Proof. For now, we are using Figure 5 as an example to prove our view and we assume the number of vaccines is one to ease of calculation. (1) We can simply vaccinate the node A in Figure 5a; (2) In Figure 5b, by using Algorithm 1, we can calculate out $\beta(A) = 0.19$ which means we can expectedly save 0.19 healthy node. At the same time, $\beta(B) = 1.71$ which is bigger than $\beta(A)$. Then, we decide to put vaccine on B for a bigger $\mathcal{V}_B$ so that we can save more nodes; (3) As shown in Figure 5c, if we are using the strategy of SVP, then we will save 0.271 max. Whereas, we are pondering whether we can get a bigger $\mathcal{V}_B$ by adding the *time division* into our strategy. That is, we wait for one time slot and do nothing with the MDL ($L1$, as shown in Figure 5c), then $G_{I_0}^{t=0}(V,E) \rightarrow G_{I_1}^{t=1}(V,E)$ and $I_0 \rightarrow I_1$. At that time, if some nodes get infected at $t = 1$, then we can surely obtain a bigger $\mathcal{V}_B$ in $L2$ as for the benefit of each node in $L2$ is much bigger than in $L1$. Otherwise, no one gets infected which means that we do not even use vaccines, but this is a small probability event. □

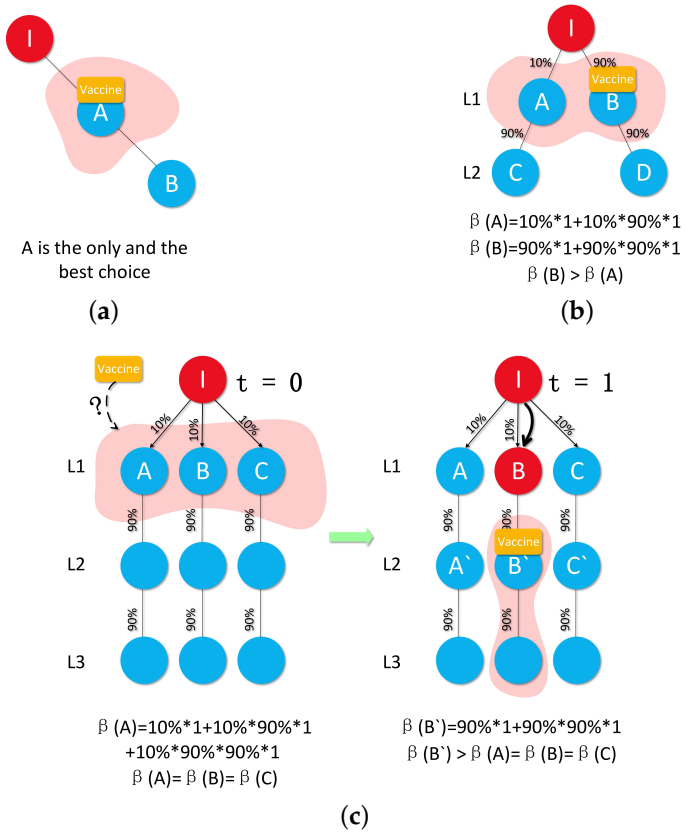


Figure 5. Examples for one vaccine: A simple case like *Simple Case 1* that we can put the only one vaccine on node A with no doubt. *Simple Case 2* is also not a hard decision for us. After comparing the benefit between A and B, we can put vaccine on B to obtain a bigger $\mathcal{V}_B$. However, in *DVP Case*, after investigating the difference of $\mathcal{V}_B$ between $t = 0$ and $t = 1$, surprisingly, we discovered $\mathcal{V}_B^{t=0} < \mathcal{V}_B^{t=1}$. That is to say, the value of $\mathcal{V}_B$ is changed by time, in addition, a bigger $\mathcal{V}_B$ can be obtained by another time. There is a big difference between waiting some time slots and without waiting. In previous assumption, we must put the $\mathcal{V}$ on $G(V,E)$ at $t = 0$ which means they could not get the bigger $\mathcal{V}_B^{t=1}$. (**a**) Simple Case 1; (**b**) Simple Case 2; (**c**) DVP Case.

4.3. Prove the Advantages of DVP(G, I_i, t) over SVP(G, I_0)

After $G(V, E)$ is converted to DT, obviously, we can get $\beta(i) > \forall\beta(j)$ so long as without taking *time dimension* into consideration, where $i \subset MDL$, $j \subset \mathbb{C}(i)$, $\mathbb{C}(i)$ is the children set of node i. However, as lemma 1 shows, once we aware that $\beta(i)$ will not always bigger than $\beta(j)$, where $j \subset \mathbb{C}(i)$, then the *time dimension* becomes an indispensable factor of VP problem. Namely, we need to take *time dimension* into our strategy to determine whether we should put vaccines at $t = \theta$ or $t = i$, where $i \subset (1, \mathcal{T})$, $i > \theta$, θ represent the current moment. So, we need to cautiously compare the expected benefit of $\mathcal{V}$ at $t = \theta$ and $t = \theta + 1$, until $k^* = 0$ or $t = \mathcal{T}$. At the same time, if we can own some bigger β at $t = \theta + 1$, then we can wait to $t = \theta + 1$ to obtain a bigger $\varphi_{G,I_0}(\mathcal{V}_{t=\theta+1})$. For better understanding, we need to look back to Figure 5c. At $t = 0$, we can have 0.271 of benefit max, as we only put vaccines on $L1$. Whereas, at $t = 1$, as have already known that node B is infected, then we can get a bigger $\varphi_{G,I_0}(\mathcal{V}_{t=1})$ as 1.71.

Lemma 2. *The benefit of each node is time-sensitive. Namely, in each propagation round, the structure of $G(V, E)$ will be changed by infected nodes and these changes directly impact the benefits.*

Proof. As for the transmission model we based on, the infected nodes set will keep developing until no one is infected in $G(V, E)$. Once one node gets recovered, it will be removed from the network and the $G(V, E)$ will not be changed either. Sometimes these changes will bring a bigger value of the benefit of $\mathcal{V}$ as compared to previous time slot, and sometimes they bring a smaller one. Meanwhile, $\beta(i) > \beta(j)$ means that immunizing i can save more healthy nodes than immunizing j on average. Thus, as $I_\theta \to I_{\theta+1}$, the value of $\varphi_{G,I_0}(\mathcal{V})$ must be changed by adjusting the placement time of vaccine, sometimes bigger and sometimes smaller. □

Theorem 2. *The strategy of obtaining the maximum value of $\varphi_{G,I_0}(\mathcal{V})$ is related to every time slot $t \subset (0, \mathcal{T})$, rather than just about $t = 0$ as previous works assumed. Moreover, the strategy of DVP(G, I_i, t) will get more benefit $\varphi_{G,I_0}(\mathcal{V})$ than SVP(G, I_0) does, as*

$$\varphi_{G,I_0}^{DVP(G,I_i,t)}(\mathcal{V}) \geq \varphi_{G,I_0}^{SVP(G,I_0)}(\mathcal{V}). \tag{4}$$

Proof. Shortly, combining the idea of lemmas 1 and 2, we know that the strategy of DVP(G, I_i, t) can save more healthy nodes. Specially, the vaccines set $\mathcal{V}$ of DVP is selected from multi time slots, only when the benefit of some candidates in $\mathcal{V}_{t=\theta+1}$ is bigger than in $\mathcal{V}_{t=\theta}$ can these candidates of $\mathcal{V}$ wait to $t = \theta + 1$. Besides, the worst case of DVP(G, I_i, t) is that there is no node has bigger benefit as compared with SVP(G, I_0). At that time, DVP(G, I_i, t) and SVP(G, I_0) will have the same $\mathcal{V}$. Thus, $\varphi_{G,I_0}^{DVP(G,I_i,t)}(\mathcal{V}) \geq \varphi_{G,I_0}^{SVP(G,I_0)}(\mathcal{V})$ is valid. □

4.4. Immunization Time Comparing

As mentioned previously, the core of our approach not only selecting out the right nodes to put vaccines on it but also choosing the right time to put it on $G(V, E)$. To do so, we need to find out that if there have $\beta(i)^{t=1} > \beta(j)^{t=0}$, where $i \subset MDL^{t=1}$ and $j \subset MDL^{t=0}$. Then, we put those nodes that has bigger β at $t = 0$ and keep others idling until $t = 1$. The computation is a cyclic operation till the $k^* = 0$, then we will get the V_B as announced at Equation (3).

The details of *Find the most valuable set of vaccines* is shown in Algorithm 2. After knowing the I_θ then we calculate out the $\mathcal{V}_{I_{\theta+1}}$, and accurately predict the $I_{\theta+1}$ then we calculate out the $\mathcal{V}_{I_{\theta+1}}$, in addition, both sets of vaccines are sorted by β. Then we can let these node wait to $t = \theta + 1$ to obtain a higher $\varphi_{G,I_0}(\mathcal{V})$.

Algorithm 2 Find the most valuable set of vaccines.

Input: $G(V,E)$, k, $P_{u,v}$, infected set I_0 and I_1, most-k-benefit vaccines set $\mathcal{V}_{I_0}$ and $\mathcal{V}_{I_1}$
Output: Vaccines set $\mathcal{V}$

1: $k^* = k$
2: **while** $I_1 \neq NULL$ and $k > 0$ **do**
3: **for** $i = 0$ **to** k **do**
4: $j = 0$
5: **if** $\mathcal{V}_{I_0}[i] \geq \mathcal{V}_{I_1}[i]$ **then**
6: Updating the $\mathcal{V}[i]$ by $\mathcal{V}_{I_0}[i]$
7: $j = j + 1$
8: **else**
9: Updating the $\mathcal{V}[i]$ by $\mathcal{V}_{I_1}[i]$
10: **end if**
11: **end for**
12: $k = k - j$
13: $I_0 \leftarrow I_1$
14: $I_1 \leftarrow I_2$
15: **end while**

As shown in Algorithm 2, it is easy to see that to get the $\mathcal{V}$ at a different time is also a major component of DVP strategy. So, we need to focus on the method of calculating those $\mathcal{V}_{t=i}$, where $i \subset (0, \mathcal{T})$. At the beginning, we proposed an accurate method, which will not incur any meaningless waiting. Then, in consideration of the performance of the algorithm, we proposed a flexible version.

Firstly, we want to know every possible status of MDL, which status can help us to analyze the benefits from waiting one time slot. For better understanding, we make an example of this calculation. We assume there are two neighbors of the infected root node of DT, A and B. So, there are four possible statuses of A and B, like AB, $\overline{A}B$, $A\overline{B}$, $\overline{AB}$, where A is the event that one node get infected and $\overline{A}$ is not.

After analyzing the possibility of each status and their benefit, we can have the expected benefit from waiting one time slot. Now, in Algorithm 3, $\mathcal{V}_B^{t=\theta}$ and $\mathcal{V}_B^{t=\theta+1}$ are corresponding benefit sets to $\mathcal{V}_{I_\theta}$ and $\mathcal{V}_{I_{\theta+1}}$. The idea of Algorithm 3 is something like '*binary decision tree*' as shown in line (5–7), by using these tools so that we can know the possibility of each event. Thus, we can use Algorithm 3 to analyze the expected benefit, and then, we can make a decision that put the nodes which have a bigger benefit at $t = \theta$ at $t = \theta$, and others wait to next time slot.

It is easy to verify that Algorithm 3 can calculate out $\mathcal{V}_B^{t=\theta}$ and $\mathcal{V}_B^{t=\theta+1}$ but with a higher computation complexity($\mathcal{O}(2^n)$). For that reason, we need to propose a fast algorithm to make it more available for a larger network. Using Monte Carlo simulation method to get the vaccines set is a well-studied method [35]. In contrast to [35], we need not find all the vaccine sets but just simulate the propagation procedure in one time slot, which has so much less complexity than [35] dose. So we adopt the idea of Monte Carlo simulation method and use it to calculate out the possible benefit of $\mathcal{V}_B^{t=\theta+1}$. At the same time, we both have $\mathcal{V}_B^{t=\theta}$ and $\mathcal{V}_B^{t=\theta+1}$, and these are all we need.

Finally, we can get the complete DWPV Algorithm 4. In Algorithm 4, we first find $\mathcal{V}_{I_0}$ and get its corresponding benefit $\mathcal{V}_B^{t=0}$, then we use the *Monte Carlo Process* to get a simulated result of $\mathcal{V}_{I_1}$ to check out that if we obtain some nodes with a higher benefit than $\mathcal{V}_{I_0}$. After that, we put those vaccines with higher benefit on $G(V,E)$ in this round and others wait to next time slot. We will keep this process going until $k^* = 0$. There is another thing that should be noticed: when we decide to put one vaccine down, there must be some neighbors of that position that have been infected or we let these vaccines wait. This little trick can bring some benefit in case some unexpected things to happen, such as we put one vaccine on the $G(V,E)$ but its neighbors are hardly infected as $G(V,E)$ is always changing.

Algorithm 3 Analyzing the benefit of each time slot.

Input: A dominator tree with infected root node, one vaccine, $P_{u,v}$, $\mathcal{V}_B^{t=\theta}$ and $\mathcal{V}_B^{t=\theta+1}$
Output: Expected benefit with one vaccine in layer 1 $\mathcal{V}_B^{t=\theta}$ and layer 2 $\mathcal{V}_B^{t=\theta+1}$

```
P_l1[ ] {Probability of transmit to layer 1 according to P_u,v}
for i = 1 to 2^N1 do
    Initialize S_all = [i] with 1 S_benefit = [i] with 0{The set of all kinds of sort in the situation of layer 1 and
    The maximum benefit will owned when one node is selected}
end for
for i = 1 to N_1 do
    Get all the immunization benefit from layer 1, namely, we get V_B^{t=θ}
end for
for i = 1 to N_1 do
    Get all the nodes infection possibility in layer 2, which are conveyed from layer 1
end for
for i = 1 to N_2 do
    Get all the immunization benefit from layer 2, namely, we get V_B^{t=θ+1}
end for
return [V_B^{t=θ}, V_B^{t=θ+1}]
```

Algorithm 4 DWPV Algorithm.

Input: $G(V,E)$, I_0 and k
Output: $\varphi_{G,I_0}(\mathcal{V})$

```
k* = k
while I_1 ≠ NULL and k* > 0 do
    G(V,E) → DT
    V_{I_0} = HeapSort(k, using Cal-Benefit() to calculate out every node's benefit of MDL_{I_0})
    Using Monte Carlo Process to get I_1
    V_{I_1} = HeapSort(k, using Cal-Benefit() to calculate out every node's benefit of MDL_{I_1})
    for i = 0 to k* do
        j = 0
        if β(V_{I_0})[i] > β(V_{I_1})[i] and neighborInfected(i) then
            Removing node i from G(V,E)
            j = j + 1
        end if
    end for
    k* = k* − j
    [G(V,E), I1] = IC-Process(G(V,E), I_0)
end while
φ_{G,I_0}(V) = IC-Process(G(V,E))
```

4.5. IC Model to SIR Model

In the beginning of our paper, we introduced two classical propagation models which could represent the process of virus transmission. In all of our paper, we illustrate our method on IC model. However, the SIR model is also important to the virus controlling problem, and it will be our future work to make our method adapted to this model. Now, we will show how to exchange from IC model to SIR model. It is obvious that SIR has a very complicated process that we can not easily

describe, so we want to follow the analyzing process of IC model and find a right place and time to put vaccines on $G(V, E)$. The difference between those two models is on $\mathcal{R}$. In IC model, we treat $\mathcal{R}$ is 1, which means each infected node just has one chance to infect its neighbors. In SIR model, $\mathcal{R} \in [0, 1]$, which means before the node is recovered, every infected node has a multi-chance to infect its neighbors. Here, we convert this multi-try infecting action into one-chance through calculating the entire possibility. Specially, the probability of being infected by one node is P and after n times trying $(n \to \infty)$ could be illustrate like $(1 - \mathcal{R})^n * (1 - p)^{n-1} * P$. Once being infected, it will be added to the infected set. The relation between the infected node and its neighbors can be represented as

$$P_{SIR} = (1 - \mathcal{R}) \times P[\frac{1}{1 - (1 - \mathcal{R}) \times (1 - P)}], \tag{5}$$

then we just need to use $P_{SIR}(u, v)$ to replace $P(u, v)$, and it will make SIR model described as IC model.

5. Experiments

In this section, we will illustrate the experiment results of our algorithm. Moreover, we use the best performance strategy (DAVA [25]) of SVP problem to be the comparison algorithm. The final results have shown that we have a better performance as compared to DAVA.

5.1. Experiments Setup

5.1.1. Real World Data Sets

In our experiments, we run the algorithms on some real-world data sets [36] (as shown in Table 2 and the main attributes of these data sets have shown in Figure 6). Every set has a different topology so that we can test our method for different scenes. The normal and the large scale of a network are both used in our experiments for comprehensive performance testing. Real-world data can testify our strategy in a detailed way, which will be really helpful for designing a realistic algorithm [37,38].

Table 2. Data set for our experiment.

Data Sets	P2P	EPINION	BRIGHTKITE	AMAZON
#Node	63K	76K	197K	335K
#Edge	180K	509K	950K	926K
#Connected Component	12	2	547	1
MAX Component	62K	76K	57K	335K

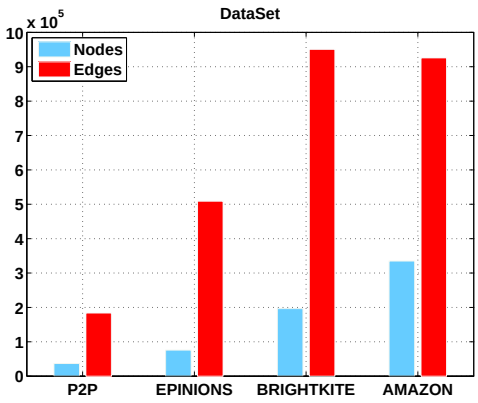


Figure 6. DataSet.

The first set, P2P, a peer-to-peer network file shared by Gnutella, each node represents a host and each edge between nodes represents the connectivity of two hosts. We use it to simulate computer virus spreading and protecting. The second set, EPINION, collected by a Website and it is a who-trust-whom social network. Each person in this set has a relationship between other nodes and the relationship relies on whether you trust him/her. This trust-based model network is well studied in following years [39]. From that network, we can examine our algorithm in the scenario that a rumor breaks out in a network and how can we stop this rumor through our method. The third set, BRIGHTKITE, a location-based social network [40], edges represent the relationship between each node like EPINION is, but the number of nodes is much bigger than EPINION which has 197K nodes rather than 76K nodes in EPINION. The last set is from AMAZON, each node represents a product of the website and also has some relationships between itself and other products. We use this set because it has a large network size, which will help us decide whether our algorithm can be used in a network of this kind of size.

5.1.2. Virus Transmission Probability

For the weight of each edge, we would like to set as $0.2, 0.4, 0.6$ as these can stand the ordinary transmission probability that happens in our usual life. Besides, different transmission probabilities can reflect the performance of our algorithm in different scenes as the transmission probability will directly affect the propagation of the virus.

5.1.3. Initial Infected Nodes Set

At the beginning of our experiments, we set the number of infected nodes as 100 and these nodes are selected at random. To better examine of our algorithm, we have changed the $|I_0|$ from 100 to 200 and 500 to see the difference that would be brought by I_0.

To get accurate results, we run our experiment 100 times to get an average result. All there experiments were conducted on a server which has an Intel Xeon E5-2660 4 cores CPU (2.20 GHz) and a 28 G RAM, the OS is 64-bit Ubuntu Server 14.04 LTS.

5.2. Results Analyzing

According to our extensive experiments, the results have shown that our method has a better benefit as compared to previous works. From those figures, we could clearly see that the performance of our algorithm is better than DAVA. When the virus transmission process is over, the last number of healthy nodes are far more than the DAVA algorithm has and DAVA has been proved as the best performance algorithm for saving nodes in graph.

5.2.1. Different Data Set

With a small data set, lowering the weight of each edge will not be of more benefit as compared to a big data set. The gap of DWPV between Figures 7a and 8a is much smaller than the gap between Figures 7d and 8d. For that reason, in a small data set, adding the number of vaccines will be a better choice. For some data sets in which nodes have a weak connectivity between other nodes like AMAZON, as shown in Figure 6, the effectiveness of putting vaccines is much less than other data sets. Under that circumstance, improving the defendable of each node will be a preferable choice. In a normal scenario, the DWPV always has more benefit than DAVA which proves the effectiveness of our solution. Especially, from Figures 7b and 8b we can see that smaller weight of edges could bring more benefit than a bigger one.

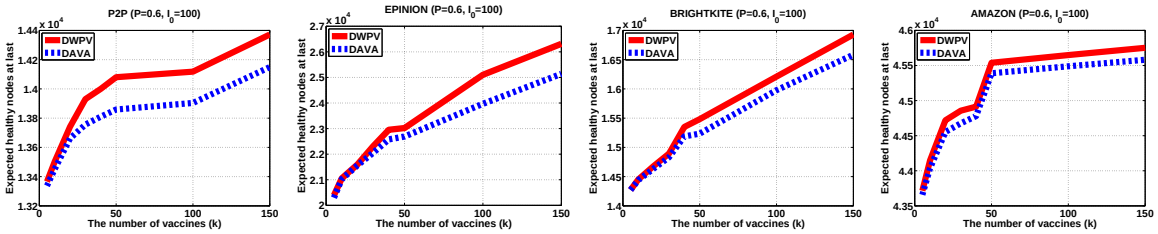


Figure 7. Experiments Results ($P = 0.6, I_0 = 100$). (**a**) P2P; (**b**) EPINION; (**c**) BRIGHTKITE; (**d**) AMAZON.

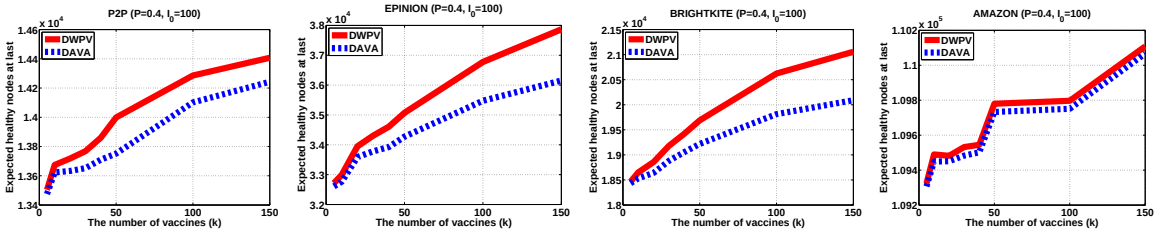


Figure 8. Experiments Results ($P = 0.4, I_0 = 100$). (**a**) P2P; (**b**) EPINION; (**c**) BRIGHTKITE; (**d**) AMAZON.

5.2.2. Different Number of Vaccines

In Figure 7, we can see that DWPV has no obvious advantage. Through our cautious observation, then we investigate whether the gap between DWPV and DAVA is up to the utility of vaccines. Specially, as we increase the number of vaccines (from 5 to 150) in Figure 7b, $\varphi_{G,I_0}(\mathcal{V})$ with SVP strategy gets a 30% elevation, but in AMAZON, there is just a 5% elevation. Thus, we believe that the performance of DWPV has a closer relationship with the utility of the vaccine. This phenomenon can also be found in Figure 7a with 5% elevation and Figure 9a with 2% elevation. It is remarkable that in Figure 9a, the DAVA and DWPV have the same result for k vaccines. As we noticed in previous sections, that is the worst case of DWPV as it has the same result as DAVA. That is also to say, the effectiveness of our method is always better than DAVA.

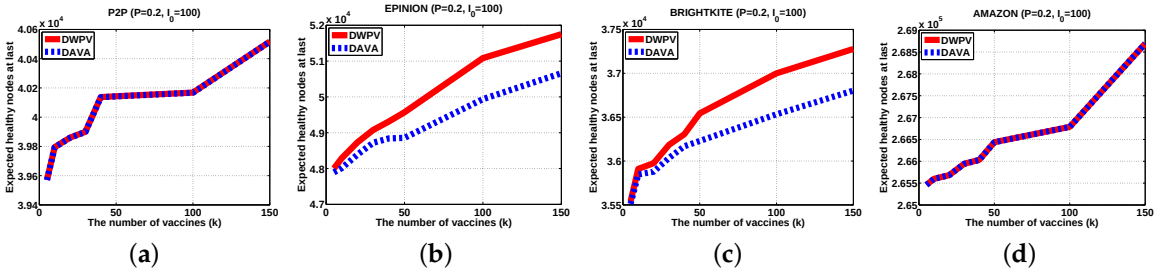
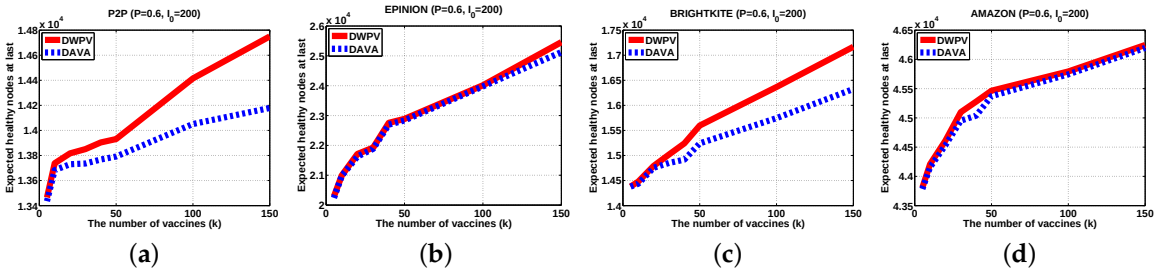


Figure 9. Experiments Results ($P = 0.2, I_0 = 100$). (**a**) P2P; (**b**) EPINION; (**c**) BRIGHTKITE; (**d**) AMAZON.

5.2.3. Different Initially Infected Node Set

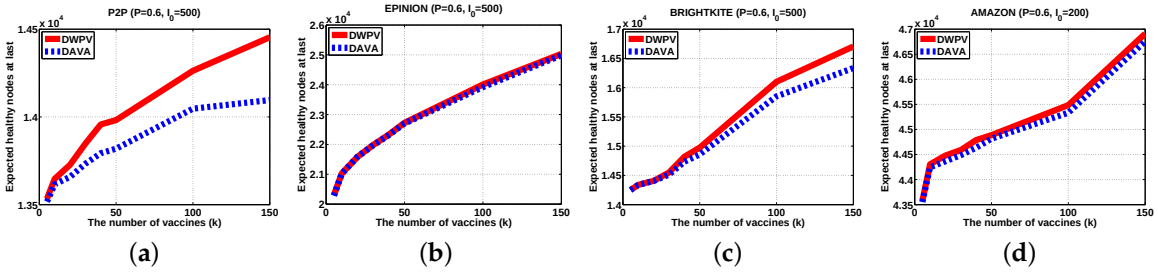
In order to verify the influence of difference I_0, we change the I_0 from 100 to 200 and 500 as shown in Figures 10 and 11. There is no doubt that our method has a better performance again, however, there are some interesting things to focus on. Generally, the increasing number of initially infected nodes will

lower the value of $\varphi_{G,I_0}(\mathcal{V})$, but in Figures 10b and 11b we have not seen this happen. That is to say, adding the number of initial nodes will not always directly affect the $\varphi_{G,I_0}(\mathcal{V})$. The point is that the utility of vaccine may be reduced by the increasing number of initially infected nodes and this will lead to the decrease of the advantage of our method, like Figure 11b.



(**a**) (**b**) (**c**) (**d**)

Figure 10. Experiments Results ($P = 0.6, I_0 = 200$). (**a**) P2P; (**b**) EPINION; (**c**) BRIGHTKITE; (**d**) AMAZON.



(**a**) (**b**) (**c**) (**d**)

Figure 11. Experiments Results ($P = 0.6, I_0 = 500$). (**a**) P2P; (**b**) EPINION; (**c**) BRIGHTKITE; (**d**) AMAZON.

In conclusion, (1) the performance of our method is always better than DAVA; (2) lower down, the transmission probability will always get a higher $\varphi_{G,I_0}(\mathcal{V})$ but this not same lower than the number I_0.

6. Related Work

The problem of node immunization has been studied for a quite long time, most of the existing works try to allocate their vaccines before they know the initially infected nodes. Wang et al. [4,5] and Tong et al. [6,7] started their works on an arbitrary graph, while Madar et al. [41] studied this problem on complex networks, like power law graphs. Cohen et al. [42] and A. L. Buchsbaum et al. [43] analyzed the *acquaintance immunization* method that the vaccinated nodes have the most number of degrees in their network, for the SIS model and the SIR model. Hayashi et al. [3] described the propagation process by using SHIR (Susceptible, Hidden, Infectious, Recovered) model. Besides, Kimura et al. [44] introduce a novel solution to minimize the spread of contamination by blocking links rather than immunize nodes. However, none of above works based on the more realistic scenario that taking the initially infected nodes into the immunization strategy.

Some more practical works have been proposed by Zhang et al. [25] and its extending work [26], in which immunization decisions are based on the known initially infected nodes. They formalize the vaccines placement problem as Data-Aware Vaccination problem and convert the original graph into a dominator tree to distribute all vaccines on the neighbors of the root immediately. However, they miss a major concept in these algorithms that the benefit of putting the vaccine on one node is the dynamic change. Hence, using these strategies would sacrifice a great number of healthy nodes which could have been saved by a multi-step distribution.

While DWPV employs an dynamic-like vaccines placement strategy, it significantly differs from all past approaches [29,45,46] in how it defines the *dynamic*. Past schemes assume the network itself is changing, and their strategies are focusing on these changes. If these changes of the network are slight, they still choose to put all vaccines on the network at once. DWPV uses a simulation-based method to estimate how the limited vaccines should be distributed over the propagation process, and the vaccines are placed in batches on the appropriate nodes and at the appropriate time. Compared with previous well studied simulation-based methods [35], DWPV differentiates itself in the scope of simulation. DWPV simulates the transmission process just around the limited scope of these candidate nodes, and we need no more simulation once all vaccines have been placed, which reduces the computation complicity to a great extent and makes our algorithm more scalable than [35].

Finally, even if the strategy has been made by DWPV, we will put the vaccines on the network until a neighbor of the designated node has been infected, which gives our algorithm a further protection of the stochastic propagation model as compared with [25,26].

7. Conclusions

In this paper, we introduce a novel method to protect healthy nodes from be infected as best as we can with limited vaccines. In contrast to previous work, we first formulate our problem to a dynamic vaccine placement problem. Then, we showed that, even if the problem is the same, there is a significantly higher benefit with the same number vaccines (up to 30%). After that, we proposed an accurate solution to help us make a decision about when to put the vaccines on the graph by calculating the benefit of waiting one time slot or not waiting. Finally, for considering the calculation time restriction, we proposed a fast algorithm for our method, which is somewhat less accurate but is time efficient (computational is linear time). The extensive experiment shows our algorithms are much better than the previous works.

Acknowledgments: This work was supported by NSFC (61272461, 61602382, 61272120, 61473109, 61422208), Science Technology Department of Shaanxi province, China (2011K0609).

Author Contributions: Chen Liu and Dan Xu conceived and designed the experiments; Wen Cui performed the experiments and analyzed the data; Shaojie Tang, Fan Wu, Xiaoqing Gong, Xiaojiang Chen, Dingyi Fang, Guihai Chen supervised this work and gave a lot suggestions to this work. All the authors joined in the paper writing.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kermack, W.O.; McKendrick, A.G. A contribution to the mathematical theory of epidemics. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **1927**, *115*, 700–721.
2. Khalil, E.B.; Dilkina, B.; Song, L. Scalable diffusion-aware optimization of network topology. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 24–27 August 2014; pp. 1226–1235.
3. Hayashi, Y.; Minoura, M.; Matsukubo, J. Recoverable prevalence in growing scale-free networks and the effective immunization. *arXiv* **2003**, arXiv:cond-mat/0305549.
4. Wang, Y.; Chakrabarti, D.; Wang, C.; Faloutsos, C. Epidemic spreading in real networks: An eigenvalue viewpoint. In Proceedings of the 22nd International Symposium on Reliable Distributed Systems, Florence, Italy, 6–8 October 2003; pp. 25–34.
5. Chakrabarti, D.; Wang, Y.; Wang, C.; Leskovec, J.; Faloutsos, C. Epidemic thresholds in real networks. *ACM Trans. Inf. Syst. Secur.* **2008**, *10*, 1.
6. Tong, H.; Prakash, B.A.; Eliassi-Rad, T.; Faloutsos, M.; Faloutsos, C. Gelling, and melting, large graphs by edge manipulation. In Proceedings of the 21st ACM International Conference on Information and Knowledge Management, Maui, HI, USA, 29 October–2 November 2012; pp. 245–254.
7. Tong, H.; Prakash, B.A.; Tsourakakis, C.; Eliassi-Rad, T.; Faloutsos, C.; Chau, D.H. On the vulnerability of large graphs. In Proceedings of the 2010 IEEE 10th International Conference on Data Mining (ICDM), Sydney, Australia, 13–17 December 2010; pp. 1091–1096.

8. Wen, Y.; Tian, X.; Wang, X.; Lu, S. Fundamental limits of RSS fingerprinting based indoor localization. In Proceedings of the 2015 IEEE Conference on Computer Communications (INFOCOM), Kowloon, Hong Kong, China, 26 April–1 May 2015; pp. 2479–2487.
9. Liu, L.; Zhang, X.; Ma, H. Optimal node selection for target localization in wireless camera sensor networks. *IEEE Trans. Veh. Technol.* **2010**, *59*, 3562–3576.
10. Liu, C.; Fang, D.; Yang, Z.; Jiang, H.; Chen, X.; Wang, W.; Cai, L. RSS Distribution-Based Passive Localization and Its Application in Sensor Networks. *IEEE Trans. Wirel. Commun.* **2016**, *15*, 2883–2895.
11. Wang, J.; Fang, D.; Yang, Z.; Jiang, H.; Chen, X.; Cai, L. E-HIPA: An energy-efficient framework for high-precision multi-target adaptive device-free localization. *IEEE Trans. Mob. Comput.* **2016**, *12*, 1–12.
12. Wang, J.; Chen, X.; Fang, D.; Wu, C.Q.; Yang, Z.; Xing, T. Transferring compressive-sensing-based device-free localization across target diversity. *IEEE Trans. Ind. Electron.* **2015**, *62*, 2397–2409.
13. Wang, G.; Zou, Y.; Zhou, Z.; Wu, K.; Ni, L.M. We Can Hear You with Wi-Fi! *IEEE Trans. Mob. Comput.* **2016**, *15*, 2907–2920.
14. Song, H.; Rawat, D.B.; Jeschke, S.; Brecher, C. *Cyber-Physical Systems: Foundations, Principles and Applications*; Morgan Kaufmann: Burlington, MA, USA, 2016.
15. Wang, Z.; Song, H.; Watkins, D.W.; Ong, K.G.; Xue, P.; Yang, Q.; Shi, X. Cyber-physical systems for water sustainability: Challenges and opportunities. *IEEE Commun. Mag.* **2015**, *53*, 216–222.
16. Squire, R.; Song, H. Cyber-physical systems opportunities in the chemical industry: A security and emergency management example. *Process Saf. Prog.* **2014**, *33*, 329–332.
17. Xiao, F.; Xie, X.; Jiang, Z.; Sun, L.; Wang, R. Utility-aware data transmission scheme for delay tolerant networks. *Peer-to-Peer Netw. Appl.* **2015**, *9*, 936–944.
18. Guo, S.; Qiang, M.; Luan, X.; Xu, P.; He, G.; Yin, X.; Xi, L.; Jin, X.; Shao, J.; Chen, X.; et al. The application of the Internet of Things to animal ecology. *Integr. Zool.* **2015**, *10*, 572–578.
19. Xiao, F.; Yang, X.; Yang, M.; Sun, L.; Wang, R.; Yang, P. Surface coverage algorithm in directional sensor networks for three-dimensional complex terrains. *Tsinghua Sci. Technol.* **2016** *21*, 397–406.
20. Wang, G.; Zhang, S.; Wu, K.; Zhang, Q.; Ni, L.M. TiM: Fine-grained rate adaptation in WLANs. *IEEE Trans. Mob. Comput.* **2016**, *15*, 748–761.
21. Zhang, Y.; He, S.; Chen, J. Data gathering optimization by dynamic sensing and routing in rechargeable sensor networks. *IEEE/ACM Trans. Netw.* **2013**, *24*, 1632–1646.
22. Zou, Y.; Wang, G.; Wu, K.; Ni, L.M. SmartScanner: Know More in Walls with Your Smartphone. *IEEE Trans. Mob. Comput.* **2015**, *15*, 2865–2877.
23. Liu, X.; Wei, T.; Liu, A. Fast Program Codes Dissemination for Smart Wireless Software Defined Networks. *Sci. Program.* **2016**, *2016*, 6907231.
24. Liu, X.; Dong, M.; Ota, K.; Yang, L.T.; Liu, A. Trace malicious source to guarantee cyber security for mass monitor critical infrastructure. *J. Comput. Syst. Sci.* **2016**, in press.
25. Zhang, Y.; Prakash, B.A. DAVA: Distributing Vaccines over Networks under Prior Information. In Proceedings of the SIAM International Conference on Data Mining (SDM 2014), Philadelphia, PA, USA, 24–26 April 2014; pp. 46–54.
26. Zhang, Y.; Prakash, B.A. Scalable Vaccine Distribution in Large Graphs given Uncertain Data. In Proceedings of the 23rd ACM International Conference on Information and Knowledge Management, Shanghai, China, 3–7 November 2014; pp. 1719–1728.
27. Kempe, D.; Kleinberg, J.; Tardos, É. Maximizing the spread of influence through a social network. In Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA , 24–27 August 2003; pp. 137–146.
28. Chen, W.; Wang, C.; Wang, Y. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, 25–28 July 2010; pp. 1029–1038.
29. Song, C.; Hsu, W.; Lee, M.L. Node Immunization over Infectious Period. In Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, Melbourne, Australia, 19–23 October 2015; pp. 831–840.
30. Zhao, C.; He, J.; Cheng, P.; Chen, J. Consensus-based energy management in smart grid with transmission losses and directed communication *IEEE Trans. Smart Grid* **2016**, *PP*, 1–13.

31. Gomez Rodriguez, M.; Leskovec, J.; Krause, A. Inferring networks of diffusion and influence. In Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, 25–28 July 2010; pp. 1019–1028.
32. Chen, W.; Wang, Y.; Yang, S. Efficient influence maximization in social networks. In Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, 28 June–1 July 2009; pp. 199–208.
33. Purohit, M.; Prakash, B.A.; Kang, C.; Zhang, Y.; Subrahmanian, V. Fast influence-based coarsening for large networks. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 24–27 August 2014; pp. 1296–1305.
34. Vinterbo, S. Privacy: A machine learning view. *IEEE Trans. Knowl. Data Eng.* **2004**, *16*, 939–948.
35. Kempe, D.; Kleinberg, J.; Tardos, É. Influential nodes in a diffusion model for social networks. In Proceedings of the 32nd international conference on Automata, Languages and Programming, Lisbon, Portugal, 11–15 July 2005.
36. Leskovec, J.; Krevl, A. SNAP Datasets: Stanford Large Network Dataset Collection. 2014. Available online: http://snap.stanford.edu/data (accessed on 7 December 2016).
37. Jiang, Y.; Song, H.; Wang, R.; Gu, M.; Sun, J.; Sha, L. Data-Centered Runtime Verification of Wireless Medical Cyber-Physical System. *IEEE Trans. Ind. Inform.* **2016**, *PP*, 1.
38. Pouryazdan, M.; Kantarci, B.; Soyata, T.; Song, H. Anchor-Assisted and Vote-based Trustworthiness Assurance in Smart City Crowdsensing. *IEEE Access* **2016**, *4*, 529–541.
39. Tang, Z.; Liu, A.; Li, Z.; Choi, Y.J.; Sekiya, H.; Li, J. A Trust-Based Model for Security Cooperating in Vehicular Cloud Computing. *Mob. Inf. Syst.* **2016**, *2016*, 9083608.
40. Tang, Z.; Liu, A.; Huang, C. Social-aware Data Collection Scheme through Opportunistic Communication in Vehicular Mobile Networks. *IEEE Access* **2016**, *4*, 6480–6502.
41. Madar, N.; Kalisky, T.; Cohen, R.; ben Avraham, D.; Havlin, S. Immunization and epidemic dynamics in complex networks. *Eur. Phys. J. B Condens. Matter Complex Syst.* **2004**, *38*, 269–276.
42. Cohen, R.; Havlin, S.; Ben-Avraham, D. Efficient immunization strategies for computer networks and populations. *Phys. Rev. Lett.* **2003**, *91*, 247901.
43. Briesemeister, L.; Lincoln, P.; Porras, P. Epidemic profiles and defense of scale-free networks. In Proceedings of the 2003 ACM Workshop on Rapid Malcode, Washington, DC, USA, 27–30 October 2003; pp. 67–75.
44. Kimura, M.; Saito, K.; Motoda, H. Minimizing the Spread of Contamination by Blocking Links in a Network. In Proceedings of the 23rd National Conference on Artificial (AAAI'08), Chicago, IL, USA, 13–17 July 2008; pp. 1175–1180.
45. Prakash, B.A.; Tong, H.; Valler, N.; Faloutsos, M.; Faloutsos, C. Virus propagation on time-varying networks: Theory and immunization algorithms. In *Machine Learning and Knowledge Discovery in Databases*; Springer: Berlin, Germany, 2010; pp. 99–114.
46. Valler, N.C.; Prakash, B.A.; Tong, H.; Faloutsos, M.; Faloutsos, C. Epidemic spread in mobile ad hoc networks: Determining the tipping point. In Proceedings of the 10th International IFIP TC 6 Conference on Networking, Valencia, Spain, 9–13 May 2011; pp. 266–280.