TB, AN, US, JPhysA/485400, 17/01/2014

IOP Publishing Journal of Physics A: Mathematical and Theoretical

J. Phys. A: Math. Theor. **47** (2014) 000000 (16pp) UNCORRECTED PROOF

On early detection of strong infections in complex networks

Yi Yu and Gaoxi Xiao

School of Electronic and Electrical Engineering, Nanyang Technological University, 50 Nanyang Ave, 639798, Singapore Q1

E-mail: egxxiao@ntu.edu.sg

Received 7 September 2013, revised 2 December 2013
Accepted for publication 2 January 2014
Published DD MMM 2014

Abstract

Various complex systems are exposed to different kinds of infections ranging from computer viruses to rumors. An intuitive solution for limiting the damages caused by such infections is to detect the infection spreading as early as possible and then take necessary actions. In this paper, we study on how much we may expect to achieve in infection control by deploying a number of monitors in complex networks for detecting the outbreak of a strong infection at its early stage. Specifically, we consider the problem of finding the optimal locations for a given number of monitors in order to minimize the worst-case infection size. The NP-hardness of the problem is proved and a heuristic algorithm is proposed. Extensive simulations on both synthetic and real-life networks show that the worst-case infection size may be put under control by deploying a moderate number of monitors in a large complex network. Effects of a few different factors, including transmissibility of the infection, network topology and probability of detection failure, are also evaluated.

Keywords: complex networks, epidemic dynamics, infection detection
PACS numbers: 89.75.−k, 89.75.Fb, 64.60.aq, 02.10.Ox

(Some figures may appear in colour only in the online journal) Q2

1. Introduction

The world is becoming more connected than ever [1] and a tremendous amount of data and information is being exchanged everyday through various complex networks, e.g., the Internet and social networks. It is well known that many of such networks are vulnerable to various types of risks, including the spreading of infections such as computer viruses, diseases and rumors [2–7]. Strong infections are usually, though not always, most dangerous in their first outbreak, when the networks have hardly any immunity or resistance to them.

 © 2014 IOP Publishing Ltd Printed in the UK

An intuitive solution for limiting the damages caused by a strong infection is to detect it as early as possible. Monitors may be installed on some network nodes and be equipped with necessary intelligence to detect the infection once it reaches the node. Upon a successful detection, an alarm can be triggered and necessary actions can be taken immediately.

Extensive studies have been done on epidemic dynamics of complex networks [8–14] and immunization schemes [15–22]. Research on developing effective monitoring schemes, however, is still rather limited.

A scheme was proposed in [23] to randomly deploy a number of 'honey-pots' as detecting agents and immunization agents in complex networks. Once any of the honey-pots is triggered by virus infection, all the honey-pots will be alerted to start the distribution of vaccination immediately. It is assumed that there exist high-speed and secured information channels, separated from the network being monitored, between all the honey-pots. All the honey-pots therefore can work together as a virtual super hub, ensuring high efficiency of vaccination distribution. The scheme is efficient yet may be of a very high cost, especially in large networks. Also it is not clear how good the random deployment of the honey-pots is compared to other possible options we may have. More closely related work was reported in [24], where the authors developed algorithms for finding the best locations of a given number of monitors to minimize the average penalties such as the population infected and detection time, etc. It was proved that the problems for these different objectives are equivalent to each other from a mathematics point of view and they are all NP-hard. Based on the results in [25], the author then proposed a heuristic algorithm named cost-effective lazy forward selection. The performance of the algorithm was evaluated in two real-life networks for multiple penalty reductions.

The work we will report in this paper has two main objectives. First, we propose an algorithm for minimizing the worst-case infection size. We argue that while controlling the average infection size is certainly of importance, minimizing the worst-case infection size may be a critical life-or-death issue for some systems: a system with high robustness in average may collapse in the worst case, though the worst case may only happen at a low probability. We prove that the problem remains to be NP-hard. A heuristic algorithm is then proposed, of which the satisfactory performance is verified by extensive simulation results. Second, we evaluate the effects of a few important factors on the effectiveness of infection detection, namely the transmissibility of the infection, network topology, and probability of detection failure. We find that (i) the worst-case infection size tends to be smaller when infection is of lower transmissibility; (ii) network topology may play a non-trivial role in affecting the effectiveness of the monitoring schemes; and (iii) in some cases, infection sizes may become significantly larger even at a low probability of detection failure. Enhancing fault tolerance of infection detection therefore remains as a challenge.

The rest of the paper is organized as follows. In section 2, we describe the system model and the assumptions adopted. In section 3, we formulate the problem, proving its NP-hardness and then propose a simple yet efficient heuristic algorithm. Simulation results on synthetic and real-life networks are presented in section 4.1. Finally, section 5 concludes the paper.

2. System model

In this paper, we adopt the well-known susceptible-infected model [9] to model the virus spreading in complex networks since in many cases, recovery may not even have started at an early stage of infection spreading. To focus on the worst case of a strong infection, we assume that time is slotted and a susceptible node adjacent to any of the infectious nodes will surely be infected at the beginning of the next time slot. A similar model has been adopted in [24].

A given number of monitors are to be deployed in the network. We assume that each of the monitors is able to detect the infection once the node on which it is installed is infected. The objective of the algorithm is to minimize the maximum number of nodes getting infected until any of the monitors triggers an alarm.

Note that in our model, we assume 100% transmissibility in the network, i.e., a susceptible node adjacent to any infected node will surely be infected in the next time slot. Not only this corresponds to the worst possible case, it may indeed to a certain extent resemble the early-stage developments of some newly-emerging strong infections, e.g., new computer virus, where hardly any nodes has immunity to the infection. It may also resemble the cascade failures in interdependent networks [7] where failures of nodes in one network surely lead to the failures of all the dependent nodes in another network, which will in turn cause more failures in the current network. The chain process of this cascade failure can be modeled as an infection spreading in a bipartite graph composed of only the inter-network links and the interdependent nodes at the ends of these links. Further extensions to the probabilistic infection cases will be discussed in section 4.1. As we will see, the proposed scheme remains to be highly effective in controlling the probabilistic infection, and the worst-case infection size tends to become smaller under weaker infection.

3. Monitor deployment algorithm

In this section we present the problem formulation and the algorithm for minimizing the worst-case infection size.

The problem can be formulated as follows. In a general graph $G(V, E)$, where V is the set of nodes and E is the set of edges, we assume that it has unit link length and equal node weight. Assume that the infection always starts from a single node and it can start from any node in the network. Let the infection have transmissibility of 1. A given number of p monitors are to be installed in the network. Once the infection reaches any of the nodes installed with a monitor (We term such nodes as *monitor nodes*.), an alarm will be triggered and the infection will be stopped immediately. The objective of the algorithm is to find the optimal locations for the p monitor nodes such that the maximum infection size (i.e., the worst-case infection size) is minimized. Note that monitor nodes themselves are not counted as being infected since we expect such nodes, thanks to the intelligence of the monitors installed on them, can resist the infection. To facilitate the later discussions, we term the above problem as the p-monitor problem.

Theorem 1. *The p-monitor problem is NP-hard.*

Proof. We first observe the vertex cover problem which is known to be an NP-complete problem [26]: given a graph $G(V, E)$ and an integer k, find a subset V_k^* of the vertices of G such that $|V_k^*| \leqslant k$ and each edge of G is incident at a vertex of V_k^*. We see that the vertex cover problem is polynomial time reducible to the p-monitor problem: if we were able to solve the p-monitor problem in general graph in polynomial time, the vertex cover problem can also be solved in polynomial time by increasing the value of p until the maximal infection size is reduced to 1. Hence, it is proved that the general p-monitor problem is NP-hard. □

3.1. The algorithm

The proposed algorithm adopts a simple iterative approach: starting with a random allocation of a given number of p monitors, we let each node join the monitor closest to itself to form

into a cluster. When there is a tie, i.e., when a node is of the same shortest distance to a few monitors, we make a copy of this node in each of these clusters (to be further explained later). Hence we shall have a total of p clusters in the network. In each cluster, we use exhaustive search to find the best location of the monitor which minimizes the maximum infection size of any infection starting from any single node in this cluster. Then all the network nodes are re-clustered, where we once again let each node join its closest monitor. The procedure is repeated until no further improvements can be made.

Note that the exhaustive search in each cluster is carried out by evaluating the maximum infection size where we let each node of the cluster serve as the monitor node in turn. We choose the one leading to the minimum worst-case infection size as the monitor node of this cluster. To simplify the algorithm and also to avoid trivial yet complicated discussions on the convergence of the algorithm, denoting the distance between an infection source and the candidate monitor node of its cluster as d, we calculate the infection size as the number of d-step neighbors of the infection source minus 1 (i.e., by ignoring the case where an infection reaches multiple monitor nodes at the same time). We claim this approximation to be legal as the proportion of the monitor nodes is small compared to the network size.

As aforementioned, when we cluster/re-cluster the network nodes and a node is found to be of the same shortest distance to several monitor nodes (Such a node is termed as a *tied* node hereafter.), we make multiple 'copies' of this node such that it appears in multiple clusters. For example, for a node n_i with the same distance to monitors $m_1, m_2, \ldots$, we let n_i belong to $C_1, C_2, \ldots$ where C_j is the cluster that contains the monitor m_j. Our experiences show that such an approach helps the algorithm achieve better results.

To allow a tied node appear in multiple clusters at the same time, however, generates an issue to be resolved: a tied node may be selected as the monitor node by multiple clusters. In our experiences, this is more likely to happen when the tied node has a high degree. Assume there are C clusters choosing the same tied node as the monitor node. A simple solution is proposed as follows: we allocate a monitor on this tied node. Then we conduct a global search to see that, among those nodes not serving as monitor nodes, which one, if installed with a monitor, will lead to the biggest reduction of the worst-case infection size. A monitor is allocated on this node and a tie, if it exists, is broken arbitrarily. Repeat the procedure for $C-1$ times until all the $C-1$ monitors are allocated.

The reason we conduct a global search to find the location for a monitor instead of a local search in a certain single cluster can be understood: when multiple clusters select the same tied node as the monitor node, usually these clusters are of relatively small sizes and the tied node has a high degree. Conducting a global search, which can be done very quickly as will be further discussed later, allows us to better balance the distribution of the monitors in the whole network. Our experiences show that the approach steadily leads to satisfactory performance.

Note that, since the infection size is calculated as the d-step neighborhood size minus 1 where d denotes the distance between the infection source and its closest monitor node, we only need to calculate d-step neighborhood sizes for each node, where d varies from 1 to a large enough value, at the beginning of the calculation once for all, and never need to repeat it again. This allows the proposed algorithm to be highly efficient in handling even extra-big network models.

The main framework of the proposed algorithm is presented below. To facilitate the discussion, hereafter we term it as the *minimizing the maximum infection* (MMI) algorithm.

Figure 1 illustrates an example of the operations of the MMI algorithm step by step. The computations of the algorithm mainly come from the one-time-only calculations of the d-step neighborhoods of each node. Note that most complex networks of research/application interest are of very short diameter [27, 28], which gives d a small-value upper bound.

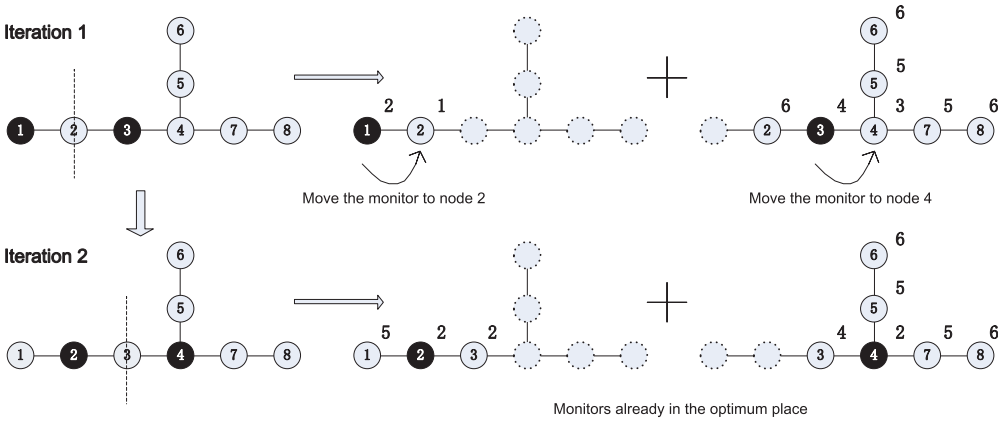

Figure 1. An example of applying the MMI algorithm to allocate 2 monitors in an 8-node network. Numbers beside the nodes show the maximum infection size where the node is selected as the monitor node and the infection starts from its own cluster. Assume that we start by randomly selecting nodes 1 and 3 as the monitor nodes. The network is then partitioned into two clusters, where node 2 appears in both clusters. Then the cluster on the left shall select node 2 as the monitor node, and the cluster on the right shall select node 4. The network will be re-clustered where node 3 appears in both clusters. No further improvement be made and the algorithm will then stop.

Algorithm 1. Minimizing the maximum infection (MMI) algorithm.

- **Initialization**
  Input the network and number of monitors p.
- **Step 1**
  Arbitrarily choose p nodes $m_1, m_2, \ldots, m_p$ as the starting p monitor nodes.
- **Step 2**
  Partition the network into p clusters by letting each nodes join its nearest monitor.
  Let tied nodes, if any, appear in multiple clusters.
- **Step 3**
  Perform exhaustive search in each cluster to find the best location for the monitor.
  Shift monitor to that node. Update m_i and get a new set of monitor nodes $m_1, m_2, \ldots, m_p$.
- **Step 4**
  If multiple monitors choose the same node, keep one monitor there and perform a global search to find the best locations for the other monitors.
- **Step 5**
  If the maximum infection size has been reduced in this iteration, go to step 2; otherwise, stop.

3.2. Convergence and evaluation

Theorem 2. *The MMI algorithm is convergent in general graphs.*

Proof. It is easy to prove the convergence of the MMI algorithm by showing that the worst-case infection size monotonically decreases in each iteration. Specifically, each iteration of the MMI algorithm is composed of two different steps: (i) re-selection of the monitor node for each cluster; and (ii) re-clustering. We examine these two steps separately.

When there is no tied node, obviously the re-selection of the monitor node in each cluster will not increase the worst-case infection size: a node will replace another node in the same cluster to serve as the monitor node if and only if such will help reduce the worst-case infection size of any infection sourced from any node in the cluster. When there exists a tied node, the

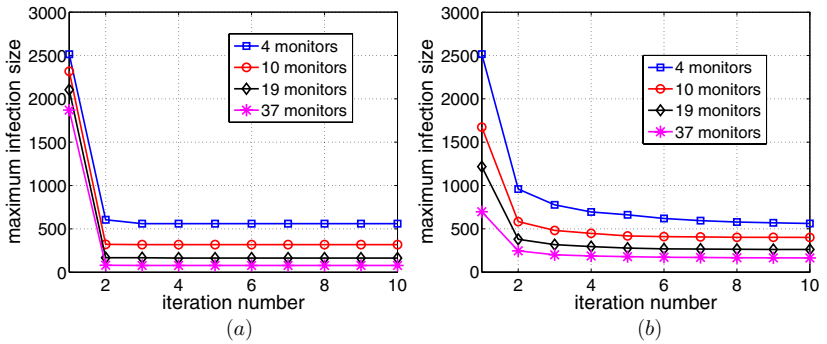

Figure 2. Worst-case infection sizes versus the number of iterations in implementing the MMI algorithm: (*a*) Internet AI; (*b*) US power grid network. The plotted curves show the average results of 100 independent implementations, each time with a different set of randomly selected initial allocations of monitors.

tied node, by serving as the monitor node, will not increase the worst-case infection size either. The global search of the monitor nodes for breaking the tie will only help make the sizes of the infections sourced from some nodes smaller.

The re-clustering operations will not increase the worst-case infection size either: the size of the infection sourced from a node is decided by the distance between this node and its closest monitor. A non-monitor node will change its cluster only if it finds a monitor node closer to itself than the previous one. The sizes of the infections sourced from those nodes which choose to change their clusters therefore will not be increased. As to those nodes choosing not to change their clusters, the sizes of the infections sourced from them are not affected in this step.

The monotonically decreasing worst-case infection size ensures that the MMI algorithm converges in finite iterations. □

Note that the convergence of the algorithm only ensures it converges to a local optima. Repeating the algorithm for a large enough times, each time with a different set of initial allocations of the monitors, helps achieve better results. The algorithm is highly efficient and can easily be repeated for many times. As later we will see, the proposed algorithm steady reaches suboptimal solutions.

Figure 2 demonstrates the fast convergence speed of the MMI algorithm. Specifically, we test on two different networks, namely the Internet autonomous system [29] and the US power grid network [30], respectively. For each testing case, we run the proposed algorithm for 100 times, each time with a different set of randomly selected initial allocations of the monitors, and keep record of how the worst-case infection size decreases along the iterations. The presented results are the average of these 100 implementations. As we can see, the MMI algorithm steadily converges in just a few iterations. We have also tested the algorithm in quite a few other networks including the ER random network, the scale-free network and many more. The MMI algorithm has always converged very quickly. **Q3**

4. Network monitoring: how hard it is?

By presenting the extensive simulation results, we will discuss on a few issues in this section, namely the performance of the MMI algorithm; the infection size under weaker infection; the effects of the network topology; and the fault tolerance of the monitoring schemes, respectively.

Table 1. The comparisons between the results of the MMI algorithm and the brute force method in different networks.

	(100,2)	(100,3)	(100,4)	(100,5)	(500,2)	(500,3)
Experiment 1: scale-free network, 100 run per graph						
Brute force	74.93	54.52	43.11	36.99	346.12	282.32
MMI algorithm	75.42	55.60	43.59	37.79	348.69	285.23
Error (in %)	0.65	1.98	1.09	2.16	0.74	1.03
Experiment 2: scale-free network, 1000 run per graph						
Brute force	76.10	54.87	43.34	37.29	343.01	280.44
MMI algorithm	76.19	55.26	43.56	37.54	343.72	280.83
Error (in %)	0.12	0.71	0.51	0.67	0.26	0.14
Experiment 3: ER random network, 1000 run per graph						
Brute force	92.77	73.06	68.85	66.04	465.52	447.50
MMI algorithm	95.56	76.29	71.40	68.65	490.50	458.11
Error (in %)	3.01	4.42	3.70	3.95	5.37	2.36

4.1. Performance of the MMI Algorithm

We first compare the results of the MMI algorithm versus the optimal solutions achieved by using the brute force method. Obviously such comparisons are only feasible in small-sized networks with a small number of monitors. The comparison results are presented in table 1. The case (m, n) means there are n monitors in a network with m nodes. For each case, we generate 100 different networks and the presented results are the average of these 100 independent implementations. We see that the MMI algorithm steadily achieves suboptimal solutions. Also we can observe that, having more runs, each time with a different set of randomly selected initial allocations of monitors, helps improve the algorithm performance. For example, for case (500, 3) in the scale-free network, increasing from 100 runs to 1000 runs helps reduce the average error (i.e., the average difference between the optimal solutions and the solutions of the MMI algorithm in the 100 different networks) from 1.03% to 0.14%.

We then proceed to conduct further performance evaluations in some bigger networks. Specifically, we adopt four different network typologies in our simulations: the BA scale-free network with 1000 nodes and 3000 edges [31]; the ER random network with 500 nodes and 2500 edges [32]; the Internet autonomous system (AS) network with 3042 nodes and 5399 edges [29]; and the US power grid network with 4961 nodes and 6594 edges [30]. These networks may closely resemble or represent the topologies of many other real-life systems [29, 30, 33–35]. While implementing the MMI algorithm on each of these networks, we conduct 1000, 5000, 100 and 100 independent runs respectively, each time with a different randomly generated initial allocations of the given number of monitors. To make comparisons, we adopt three benchmark algorithms: *random deployment* which randomly distributes the monitors in the network; *hub occupation* which installs the monitors on highest-degree nodes, and greedy algorithm developed in [24] for minimizing the average infection size (For convenience, hereafter we term it as the *minimizing average infection* (MAI) algorithm.). It is of research interest to see how big the worst-case infection size is when we implement a monitoring scheme for minimizing the average infection size, and vice versa. For the random deployment, we randomly generate 1000 different allocations and choose among them the one leading to the best performance. For the BA and ER network models, we randomly generate ten different networks for each of them and show the average results on the ten networks with 95%

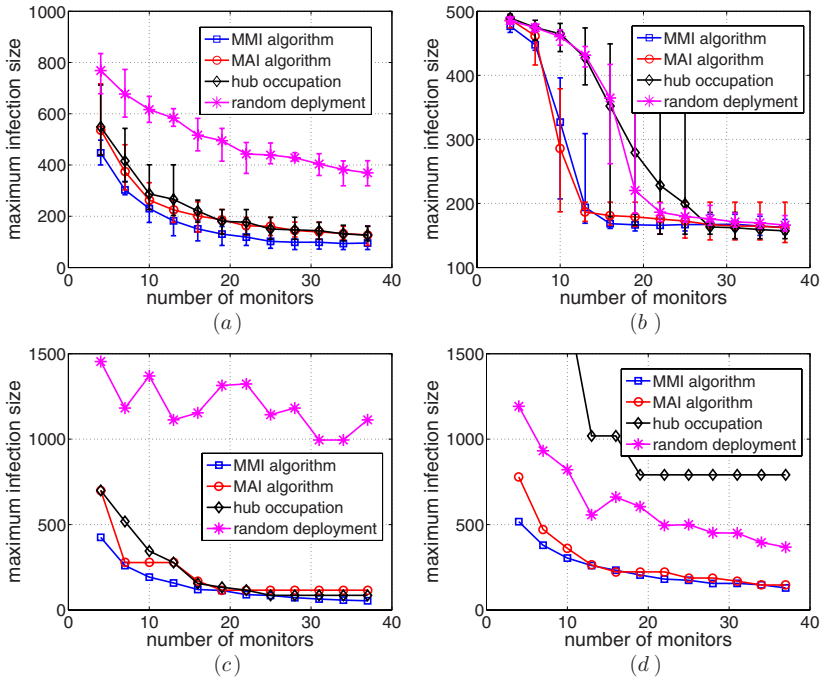

Figure 3. Maximum infection size versus the number of monitors deployed in (a) scale-free network; (b) ER random network; (c) Internet AI; and (d) US power grid network.

confidence interval. For all the network models, we compare the average and the worst-case infection sizes while implementing each of the four different algorithms.

Figures 3 compares the worst-case infection sizes in different networks when we adopt different algorithms. We see that the MMI algorithm steadily achieves the best performance when the number of monitors is reasonably large. With a moderate number of monitors, e.g., 25 monitors in a 1000-node scale-free network, the worst-case infection size would be about 10% of network size. Further improvements, however, have to come at a much higher cost: doubling the number of monitors to 50 only slightly decreases the worst-case infection size to about 8% of the network size.

Other interesting observations include: (i) hub occupation does not lead to the best performance, even in the scale-free network. High-degree hubs are not necessarily always the best choices for monitor installation; (ii) algorithms other than the MMI algorithm tend to have a large fluctuation range in their results on the ER random network. This may not be a surprise since the ER random network does not have any obvious hubs or 'intuitively favorable choices' for monitor installation; (iii) the two real-life networks, large as they are, are actually easier to be effectively monitored compared to the homogeneous BA and ER networks. Though it is difficult to prove in mathematics, the existence of the community structures in the network appears to helpin making the monitoring job easier.

Figure 4 compares the performances of the different algorithms in controlling the average infection size. It is not a surprise that the MAI algorithm is usually the winner; but the MMI algorithm is never far away when the number of monitors is reasonably large. This is good news for us: the effective control of worst-case scenario does not have to come at a high cost of sacrificing the average performance.

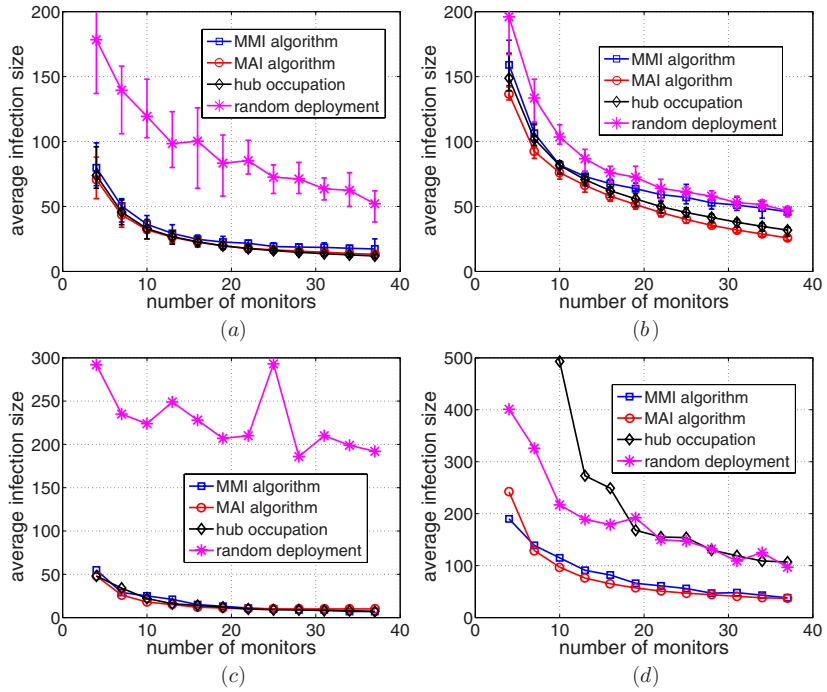

Figure 4. Average infection size versus the number of monitors deployed in (*a*) scale-free network; (*b*) ER random network; (*c*) Internet AI; and (*d*) US power grid network.

Other interesting observations include: (i) it is relatively easy to put the average infection size under control when the network is scale-free or closely resembles a scale-free network (e.g., the Internet AS model). For example, installing 25 monitors in the BA network suppresses the average infection size to be about 2% of network size; having 37 monitors in the Internet AS network leads to an average infection size of only seven nodes. Controlling the average infection size in the ER random network, however, appears to be more difficult: installing 37 monitors in a 500-node network, the average infection size is still about 30; (ii) the two real-life networks perform differently. As aforementioned, it is easy to control the average infection size in the Internet AS network. In the power grid network, however, it is significantly more difficult: installing 80 monitors by using the MAI algorithm, the average infection size can still be as large as 31. This is mainly due to the different network diameters it has. With an average shortest-path distance of 18.99 between every pair of nodes, it takes a large number of monitors to suppress the average infection size to be very small in the US power grid network.

4.2. Infection size of probabilistic spreading

We now test on the performance of the proposed scheme under probabilistic spreading, or in other words, under weaker infection with transmissibility $p < 1$. Weaker infection spreads out more slowly. But there is a (low but nonzero) chance that, instead of reaching the closest monitor and triggering an alarm, a weaker infection may infect almost the whole network without touching any monitor node (unless the monitor nodes form into a cut set which, once getting removed, will leave the network be broken into pieces). Discussing on the 'worst-case'

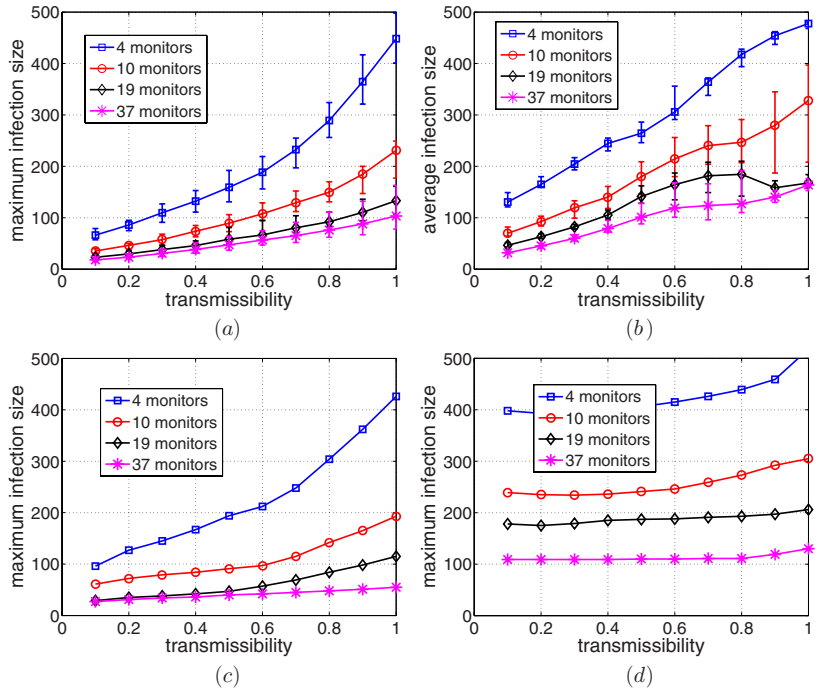

Figure 5. Expected maximum infection size versus the transmissibility in (a) scale-free network; (b) ER random network; (c) Internet AI; and (d) US power grid network.

infection size under such rare event may not be very meaningful. Instead, we will evaluate the 'expected' maximum and average infection sizes.

The 'expected' maximum and average infection sizes under probabilistic spreading are defined as follows: for each network node, run 100 independent simulations of infection starting from this node. Term the average infection size of these 100 independent implementations as the expected infection size of this node. The expected maximum infection size of the network is defined as the maximum of each network node's expected infection sizes; and the expected average infection size of the network is the average of all the nodes' expected infection sizes.

We still test on the four types of networks with the same parameters as those in section 4.1. For the BA and ER networks, we each generate ten different networks. Once again the 95% error bar is given for the results of the ten networks.

Figures 5 and 6 illustrate the relationship between the infection transmissibility and the expected maximum/average infection size of the network, respectively. The results indicate a clear trend that a smaller value of p leads to a smaller expected maximum/average infection size. For example, with four monitors in a 1000-node scale-free network, the expected maximum infection size can be reduced from 426 to 66.2 when transmissibility decreases from 1 to 0.1. The trend, however, is least obvious in the US power grid network: the large diameter and low nodal degree of the network make it less sensitive to changes in infection transmissibility.

4.3. Effects of network topology on effectiveness of the monitoring schemes

Figures 3 and 4 to an extent have already showed the effects of network topology. The most notable conclusions include: (i) it is relatively easy to put the average infection size under

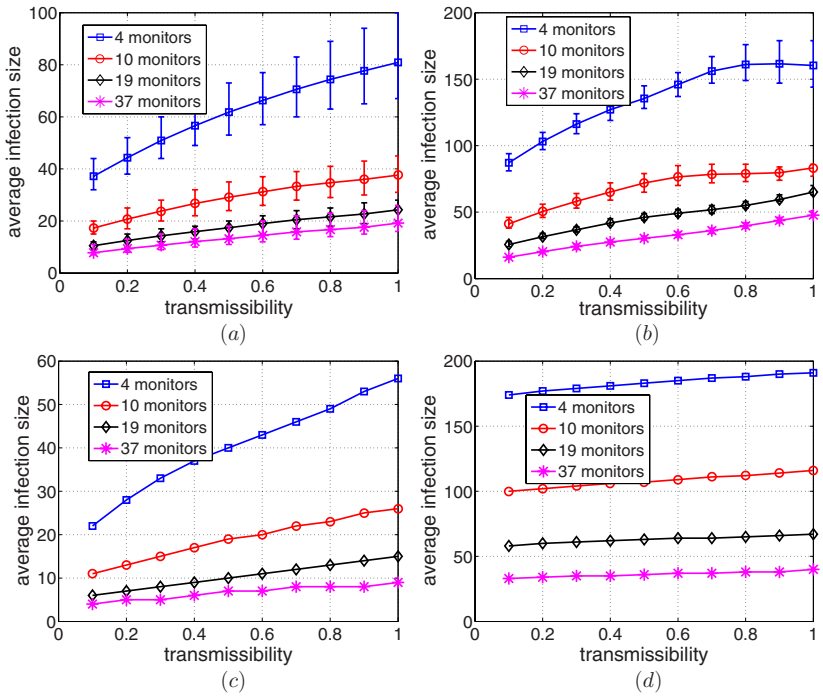

Figure 6. Expected average infection size versus the transmissibility in (*a*) scale-free network; (*b*) ER random network; (*c*) Internet AI; and (*d*) US power grid network.

control in scale-free network but more difficult in networks without obvious hubs or with a relatively large diameter; (ii) a moderate number of monitors can suppress the worst-case infection size to be reasonably small. To have a very small worst-case infection size (e.g., 1%–2% of network size), however, requests a very large number of monitors. Such conclusions are useful, but they only reflect the expected and worst-case behaviors of the networks. In this section, we shall have a look at the distribution of the infection size for infection starting from different network nodes.

The assumptions/parameters adopted in this part of simulations are summarized as follows. We still consider all the four network models as stated in section 4.1. In each network, we let each of the network nodes (except for the monitor nodes) serve as the infection source in turn and keep a record of the corresponding infection size. Such records allow us to plot the distribution of the infection size in each network. For the BA and ER network models, we still randomly generate ten different networks and for each of them, keep a record of the distribution of infection size. The 95% error bars for both network models are rather small and therefore are not plotted in the figures. We assume that ten monitors are installed in each network by using the MMI algorithm.

Figures 7 illustrates the distributions of infection size in the four networks respectively. We see that in the BA network and the Internet AS network, in most cases the infection size is very small. Large infection size only occurs at a very low probability. In the ER random network, usually the infection size is small; but there exists a non-trivial proportion of large infection-size cases, making the average infection size in this network difficult to be controlled. The worst case happens in the US power grid network where a significant proportion of all the cases lead to a relatively large infection size. The lesson is that it is difficult to monitor the

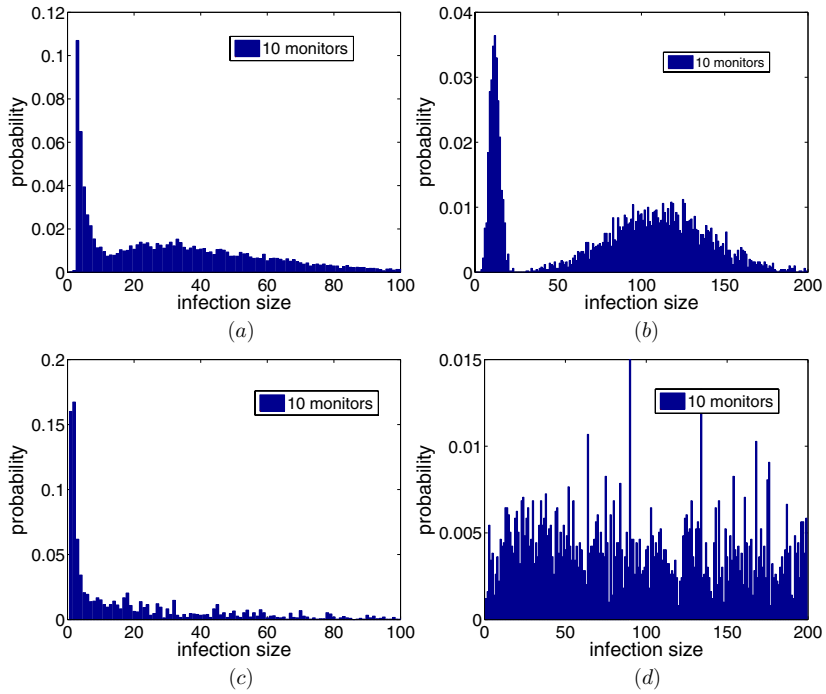

Figure 7. Probability distribution of infection size for infections sourced from each different nodes of the (*a*) scale-free network; (*b*) ER random network; (*c*) Internet AI; and (*d*) US power grid network. The average infection sizes are 36.3, 81.9, 25 and 115 respectively.

infection spreading at its early stage in networks with large diameters: infection spreading in such networks may not have a very large worst-case infection size, but the average infection size tends to be big.

4.4. Where are the monitors allocated?

In this section, we briefly examine on the statistical properties of the best monitor allocations.

An easy observation is that, in scale-free networks with short diameters, the monitor nodes selected by the MMI/MAI algorithms tend to be hub nodes. For example, in ten randomly generated BA model networks deployed with 37 monitors, MMI and MAI algorithms averagely select 19.8 and 28.2 of the first 37 highest-degree hubs as the monitor nodes respectively. Rank the hubs in a decreasing order of their degrees, the average degree of the highest-degree hub that is *not* selected as a monitor node is 11.6 and 22.5 for MMI and MAI algorithms respectively. In the Internet AS model deployed with 37 monitors, MMI and MAI algorithms hit 27 and 20 of the first 37 highest-degree hubs respectively; and the average degrees of the highest-degree node that is not selected as a monitor node are 22 and 15 for MMI and MAI algorithms respectively. Figure 8(*a*) shows in the Internet AS model, the degrees of the monitor nodes in a descending order while using MMI, MAI and hub occupation methods, respectively. The results show that most of the highest-degree nodes are selected as monitor nodes. The situation is basically the same in the BA model.

Such an observation can be easily understood: in such networks, infection starting from any node usually reaches a hub node within one or two steps. It is important that infection does

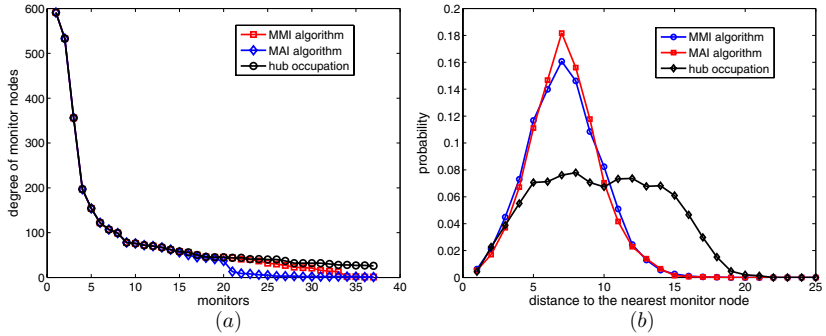

Figure 8. Statistical properties of the selected monitor nodes: (*a*) Degrees of monitor nodes in a descending order in the Internet AS model network; (*b*) Distribution of distances between infection sources and their nearest monitors in US power grid network.

not have a chance to take down a high-degree hub and its large number of adjacent nodes. We call this *degree effect*.

From figure 8(*a*), however, we can also observe that sometimes the selected monitor nodes can be of rather low degrees. It would be of interest to figure out why such nodes are selected. We conducted a detailed study on these low-degree nodes and found that they normally have at least a 2-hop distance from the nearest hub node. This means that, if monitors are only installed on hub nodes, infection starting from any of these low-degree nodes shall have a chance to propagate two steps, infecting its 2-hop neighborhood area which could be actually quite large, before it is stopped. For example, in one case of using the MMI algorithm to deploy 22 monitors in the BA model network, there are five monitor nodes with degrees lower than 10. Their degrees are 3, 3, 4, 5, 7 respectively. If monitors are installed on the first 22 highest-degree nodes and let infection start from each of these low-degree nodes, the infection sizes would be 32, 36, 24, 46 and 46 respectively. This explains why these nodes are selected: their special locations of being relatively far away from any of the hub nodes make them dangerous candidates of infection source. Hence these nodes have to be closely monitored.

In US power grid network, we have yet some other interesting observations: while deploying 37 monitors, MMI and MAI algorithms hit 7 and 0 of the 37 largest-degree hubs respectively, and the largest hub is not selected by any of the two methods. The main reason, as we believe, is that in networks with very large diameters, the monitor nodes need to have a geographically-balanced distribution: in such networks, the infection size is largely decided by the distance the infection can travel before it is stopped, rather than the degrees of the nodes it has a chance to infect. We call such the *distance effect*.

To support the above argument, figure 8(*b*) illustrates the distribution of the distances between the infection sources and their nearest monitors when we install 37 monitors in the US power grid network and then exhaustively let each of the network nodes serve as the infection source. We see that the MMI/MAI algorithms achieve a much shorter distance between the monitors and infection sources. The worst-case distance is 15 hops while by using the hub occupation method, the longest distance is 20 hops. The average distances between the infection sources and their closest monitors achieved by the three algorithms are 11.40, 10.54 and 14.38 respectively.

The BA model and the US power grid network provide two extreme cases where the degree effect and the distance effect dominate respectively. In many real-life networks with different specific network architectures, the worst/average infection size is influenced by a

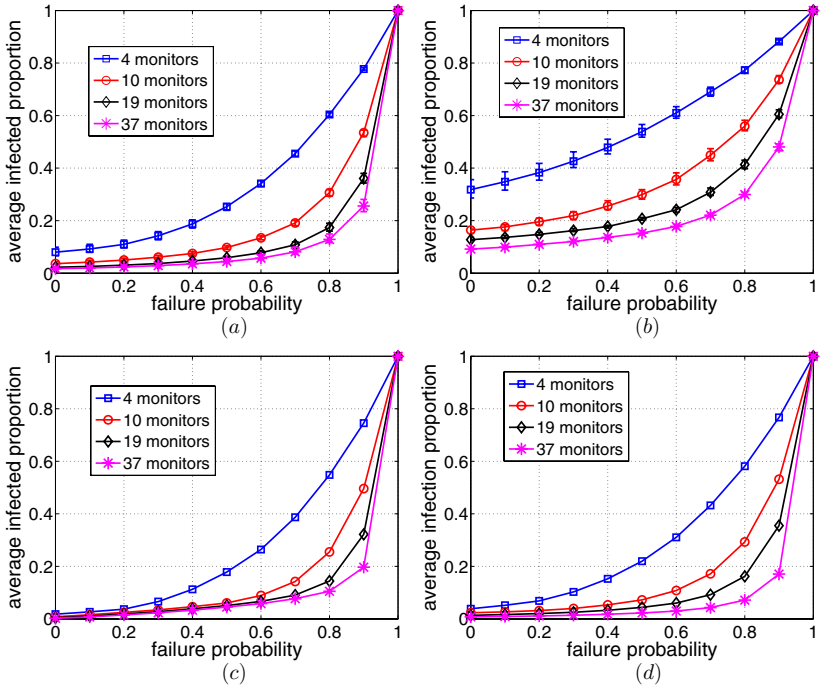

Figure 9. Average infection size versus the probability of detection failure while using the MMI algorithm in (*a*) scale-free network; (*b*) ER random network; (*c*) Internet AI; and (*d*) US power grid network.

combination of these two effects. Finding the best allocations of the monitor nodes therefore essentially requires an algorithmic approach.

4.5. Fault tolerance of the monitoring schemes

Up till now, we have been assuming that every monitor never fails to trigger an alarm once the infection reaches it. This may be an over-optimistic assumption compared to real-life cases: while it may be indeed easy to detect cascading failure in a power grid or rumor spreading in social networks, spreading of a brand-new, well-disguised computer virus may stand a good chance to skip the attention of a few, or even many, monitors. Another example may be the spreading of a new virus in human society. It is, therefore, important to examine and understand the fault tolerance of the monitoring schemes. In this section, we evaluate the average infection size when each monitor has a certain probability of failing to trigger an alarm (Note that the worst-case infection size under such case is always 'the whole network', though such may only happen at an extremely low probability.).

Figure 9 illustrates the average infection size versus the probability of the detection failure in the four networks. All the parameters remain the same as those in section 4.1 unless otherwise specified. For each network, we run ten rounds of independent simulations. The error bars for the results on the BA and ER network models are for simulations on the ten different networks. We plot in each figure a few curves corresponding to different cases with different numbers of monitors respectively.

We can easily observe that in all the networks, the performance of the network monitoring is rather sensitive to the detection failure. For example, in the Internet AS network with 37

monitors, the average infection size is only 7 when there is no detection failure. When the probability of detection failure is increased from 10% to 50% at a step of 10%, the average infection size is increased to 23, 46, 71, 101 and 137 respectively. Note that the average infection size with only four monitors and 100% successful rate is 55, having 37 monitors with an 80% successful rate in detection, averagely speaking, is only as good as having four monitors with 100% successful rate. Even a low detection failure rate can seriously compromise the effectiveness and usefulness of the monitoring schemes.

Note that the above observations still hold when we adopt the MAI algorithm, hub occupation or random method to decide the locations of the monitors. Enhancing the fault tolerance of the monitoring schemes therefore remains as a challenge which probably requests a different approach in algorithm design.

5. Conclusion and future work

In this paper, we studied on monitor deployment in complex networks for early-stage detection of a strong infection. Specifically, we considered the problem with the objective of minimizing the worst-case infection size. The NP-hardness of the problem was proved and a simple heuristic algorithm was proposed. Extensive simulation results showed that, generally speaking, the average and the worst-case infection size may be put under control by installing a moderate number of monitors in a large network. Monitoring weaker infection for damage control, averagely speaking, tends to be easier. It was also revealed that network topology may have non-trivial influences on the effectiveness of infection detection. The easy-to-be-infected scale-free networks are also relatively easy to be monitored. A challenge remained, however, is to enhance the fault tolerance of the monitoring schemes; otherwise, in some cases a low detection failure rate may lead to serious losses.

Our future work would naturally be focusing on developing fault-tolerant monitoring schemes and understanding the costs such schemes may have to impose. Another interesting topic is to investigate the effects of the community structures on infection detection and control in complex networks.

Acknowledgments

The work is partially supported by Ministry of Education, Singapore, under the contract no. RG69/12.

References

[1] Barabási A-L 2003 *Linked: How Everything is Connected to Everything Else and What it Means for Business, Science, and Everyday Life* reissue edn (Plume)
[2] Crucitti P, Latora V, Marchiori M and Rapisarda A 2004 Error and attack tolerance of complex networks *Physica* A **340** 388–94
[3] Lloyd A L and May R M 2001 How viruses spread among computers and people *Science* **292** 1316–7
[4] Shah D and Zaman T 2011 Rumors in a network: who's the culprit? *IEEE Trans. Inform. Theory* **57** 5163–81
[5] Albert R, Albert I and Nakarado G L 2004 Structural vulnerability of the north american power grid *Phys. Rev.* E **69** 025103
[6] Liljeros F, Edling C R, Núñez Amaral L A, Stanley H E and Åberg Y 2001 The web of human sexual contacts *Nature* **411** 907–8
[7] Buldyrev S V, Parshani R, Paul G, Eugene Stanley H and Havlin S 2010 Catastrophic cascade of failures in interdependent networks *Nature* **464** 1025–8

[8] Moore C and Newman M E J 2000 Epidemics and percolation in small-world networks *Phys. Rev.* E **61** 5678–82
[9] Bailey N T J 1975 *The Mathematical Theory of Infectious Diseases and its Applications* vol 66 (London: Griffin)
[10] Pastor-Satorras R and Vespignani A 2001 Epidemic spreading in scale-free networks *Phys. Rev. Lett.* **86** 3200–3
[11] Newman M E J and Watts D J 1999 Scaling and percolation in the small-world network model *Phys. Rev.* E **60** 7332–42
[12] Zhou J, Xiao G, Cheong S A, Fu X, Wong L, Ma S and Cheng T H 2012 Epidemic reemergence in adaptive complex networks *Phys. Rev.* E **85** 036107
[13] Wang Y and Xiao G 2012 Epidemics spreading in interconnected complex networks *Phys. Lett.* A **376** 2689–96
[14] Pastor-Satorras R and Vespignani A 2002 Epidemic dynamics in finite size scale-free networks *Phys. Rev.* E **65** 035108
[15] Cohen R, Havlin S and ben Avraham D 2003 Efficient immunization strategies for computer networks and populations *Phys. Rev. Lett.* **91** 247901
[16] Pastor-Satorras R and Vespignani A 2002 Immunization of complex networks *Phys. Rev.* E **65** 035104
[17] Dezső Z and Barabási A-L 2002 Halting viruses in scale-free networks *Phys. Rev.* E **65** 055103
[18] Chen Y, Paul G, Havlin S, Liljeros F and Eugene Stanley H 2008 Finding a better immunization strategy *Phys. Rev. Lett.* **101** 058701
[19] Wang Y, Xiao G, Hu J, Cheng T H and Wang L 2009 Imperfect targeted immunization in scale-free networks *Physica* A **388** 2535–46
[20] Wang C, Knight J C and Elder M C 2000 On computer viral infection and the effect of immunization *ACSAC'00: 16th Annu. Computer Security Applications Conf.* pp 246–56
[21] Gomez-Gardenes J, Echenique P and Moreno Y 2006 Immunization of real complex communication networks *Eur. Phys. J.* B **49** 259–64 **Q6**
[22] Echenique P, Gómez-Gardeñes J, Moreno Y and Vázquez A 2005 Distance-*d* covering problems in scale-free networks with degree correlations *Phys. Rev.* E **71** 035102
[23] Goldenberg J, Shavitt Y, Shir E and Solomon S 2005 Distributive immunization of networks against viruses using the 'honey-pot' architecture *Nature Phys.* **1** 184–8
[24] Leskovec J, Krause A, Guestrin C, Faloutsos C, VanBriesen J and Glance N 2007 Cost-effective outbreak detection in networks *KDD'07: Proc. 13th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining* pp 420–9
[25] Nemhauser G L, Wolsey L A and Fisher M L 1978 An analysis of approximations for maximizing submodular set functions *Math. Program.* **14** 265–94
[26] Karp R M 1972 Reducibility among combinatorial problems *Complexity of Computer Computations* (*The IBM Research Symposia Series*) ed R E Miller, J W Thatcher and J D Bohlinger **Q7**
[27] Bollobás B and Riordan O 2004 The diameter of a scale-free random graph *Combinatorica* **24** 5–34
[28] Albert R, Jeong H and Barabási A-L 1999 The diameter of the world wide web *Nature* 130–1 **Q8**
[29] Leskovec J, Kleinberg J and Faloutsos C 2005 Graphs over time: densification laws, shrinking diameters and possible explanations *KDD'05: Proc. 11th ACM SIGKDD Int. Conf. on Knowledge Discovery in Data Mining* pp 177–87
[30] Watts D J and Strogatz S H 1998 Collective dynamics of 'small-world' networks *Nature* **393** 440–2
[31] Barabási A-L and Albert R 1999 Emergence of scaling in random networks *Science* **286** 509–12
[32] Erdös P and Rényi A 1959 On random graphs: I *Publ. Math.* **6** 290–7
[33] Barabási A-L, Albert R and Jeong H 2000 Scale-free characteristics of random networks: the topology of the world-wide web *Physica* A **281** 69–77
[34] Faloutsos M, Faloutsos P and Faloutsos C 1999 On power-law relationships of the internet topology *SIGCOMM Comput. Commun. Rev.* **29** 251–62
[35] Newman M E J, Strogatz S H and Watts D J 2001 Random graphs with arbitrary degree distributions and their applications *Phys. Rev.* E **64** 026118

QUERIES

Page 1

Q1

Author: Please check whether the affiliation address is okay as included.

Q2

Author: Please be aware that the color figures in this article will only appear in color in the Web version. If you require color in the printed journal and have not previously arranged it, please contact the Production Editor now.

Page 6

Q3

Author: Please expand the acronyms 'ER' and 'BA', if required.

Page 15

Q4

Author: Please check the details for any journal references that do not have a blue link as they may contain some incorrect information. Pale purple links are used for references to arXiv e-prints.

Q5

Author: Please check whether the book title is okay as amended in reference [1] .Also provide the location of the publisher in the same.

Page 16

Q6

Author: Please check whether the year is okay as included in reference [21].

Q7

Author: Please provide the publisher details (city and name) in reference [26].

Q8

Author: Please provide volume number in reference [28].