Dismantling efficiency and network fractality

Yoon Seok Im and B. Kahng*
CCSS, CTP and Department of Physics and Astronomy, Seoul National University, Seoul 08826, Korea

(Received 24 April 2018; revised manuscript received 7 July 2018; published 26 July 2018)

In network dismantling, a minimal set of nodes is identified whose removal breaks the network into small components of subextensive size. Because finding the optimal set of nodes is an NP-hard problem, several heuristic algorithms have been developed as alternative methods, for instance, the so-called belief propagation-based decimation (BPD) algorithm and the collective influence (CI) algorithm. Here, we test the performance of these algorithms and analyze them in terms of the fractality of the network. Networks are classified into two types: fractal and nonfractal networks. Real-world examples include the World Wide Web and the Internet at the autonomous system level, respectively. They have different ratios of long-range shortcuts to short-range ones. We find that the BPD algorithm works more efficiently than the CI algorithm no matter whether a network is fractal or not. On the other hand, the CI algorithm works better on nonfractal networks than on fractal networks. We construct diverse fractal and nonfractal model networks by controlling parameters such as the degree exponent, shortcut number, and system size and investigate how the performance of the two algorithms depends on structural features.

DOI: 10.1103/PhysRevE.98.012316

I. INTRODUCTION

Network science concerns phenomena in systems of multiple nodes interacting with each other through links. In heterogeneous networks, whose nodes have various numbers of connections and types of hierarchy, identification of influential nodes is an important issue. If we know the characteristics of nodes that play crucial roles in a specific dynamic process, then we can control the process by modifying the connectivity of the network. For instance, one may wonder how to identify supertransmitters, who are likely to induce an epidemic outbreak when they are infected. Once the supertransmitters are identified, the spread of disease can be suppressed by vaccinating or quarantining them first. Because disease epidemics can spread rapidly on networks and be fatal to humans, many studies have been performed to identify supertransmitters using epidemic models such as the susceptible-infected-recovered model [1–6]. As in the prevention of epidemics, modification of a network requires resources and time. Thus, optimization of influence parameters by selecting an appropriate set of nodes is essential when propagation processes need to be controlled on complex networks. Node selection can be used in any attempt to control the behavior of an entire network using limited resources, such as viral marketing [7], political campaigns, and military intelligence [8].

In optimal percolation, also called network dismantling, a minimal set of nodes is identified whose removal breaks the giant connected component of a network into small components of subextensive size. It can be mapped to optimal immunization and the spreading problem [9]. Practically, optimal percolation offers a general countermeasure against infectious disease, no matter how contagious it is, where the size of the giant component is an upper bound on epidemic outbreak [10]. Optimal percolation is an NP-complete problem, like the other optimal influence problems [11]. One cannot expect a deterministic algorithm to work within polynomial-time complexity unless the answer to the famous P-NP problem is proved to be affirmative [12]. Instead, the solution can be chosen from among the candidates by guessing and then checked in polynomial time. Given a finite fraction as the number of nodes to be removed, we can find the exact optimal percolation set with the given size by checking every possible candidate. However, as the system size increases, the number of cases increases exponentially. Thus, to deal with large networks, we need to develop a method to guess a good candidate solution at a scalable time complexity.

As the first step, one can delete important nodes in turn, using centrality measures such as the degree, eigenvector centrality, or closeness centrality as a criterion for the importance [13]. For many graph instances, the performance can be enhanced by recalculating the centrality measures after each removal [14]. However, a centrality measure does not guarantee the importance of a collection of nodes as a set, even if each chosen node is important by itself. Even if we choose a good centrality measure while trading off appropriately between the scalability and accuracy, its effectiveness depends heavily on the topology of the target network. Traditional optimization methods such as the Monte Carlo method take a very long time to approach the optimal value, and greedy algorithms often give unreliable results [15].

As alternatives, several heuristic algorithms theoretically based on belief propagation (BP) [16] have been proposed recently, including those designed to find the dismantling set, that is, the minimal set of nodes that can be removed to break the giant connected component into small pieces of subextensive size. In BP, global information is transmitted by iteration of message-passing equations for local quantities. This characteristic is appropriate for the optimal percolation problem, where we should consider the global influence of

*Email address: bkahng@snu.ac.kr

node removal, although we have to keep the quantities local because we need a scalable algorithm. In fact, most state-of-the-art algorithms use a BP method based on the spin glass theory in statistical physics and some characteristics described by graph theory. The BP-based decimation (BPD) algorithm [17] shows outstanding performance among heuristic algorithms for most graph instances. The min-sum algorithm [18], the performance of which is known to be comparable to that of the BPD algorithm, is also based on BP. The collective influence (CI) algorithm [9], one of the algorithms based on centrality, also starts theoretically by considering the stability of message-passing equations, although it is approximated to use a centrality measure, the so-called CI. The three algorithms mentioned above are very scalable and are known to work in $\mathcal{O}(N \log N)$ time complexity.

The optimal percolation problem is deeply related to the characteristics of loops. Many dismantling algorithms assume that the network is locally treelike, that is, the number of local loops is negligibly small. Under this assumption, it can be shown that the minimal dismantling set coincides with the minimal decycling set, that is, the minimal set of nodes whose deletion leads to the removal of every loop in the graph [18]. Those algorithms in Refs. [17,18] use this fact as a first step in dismantling: First, remove loops from the network, and then break down the remaining trees into small pieces. Furthermore, the validity of the BP method relies on the loop characteristics of the network. The BP method is exact on tree graphs, and it gives a good approximation if the correlation between neighbors of a node is sufficiently small in the cavity graph [19]. This condition is realized if there is no local loop, i.e., the network is locally treelike. However, algorithms based on the BP method are reportedly still effective on real-world networks that contain many local loops [17,18,20].

The loop characteristics of networks have been categorized quantitatively by the fractal scaling property [21]. Fractal scaling represents the power-law relation between the minimum number of boxes N_B to cover the entire network and the size of the boxes ℓ_B, $N_B(\ell_B) \sim \ell_B^{-d_B}$, with a finite fractal dimension d_B [22]. It has been observed, however, that not all networks are fractals, and most of the random network models proposed to date are also not fractals. Here we aim to characterize the effectiveness of dismantling algorithms on loopy graphs in terms of the fractality of networks. We find that the BPD algorithm works more efficiently on fractal networks than on nonfractal networks, whereas the CI algorithm is better on nonfractal networks. Moreover, the difference between the performances of the two algorithms is smaller for nonfractal networks than for fractal networks.

This paper is organized as follows. We show the dismantling performance of the BPD and CI algorithms on two real-world networks, the World Wide Web and the Internet at the autonomous level, in Sec. II. Similar work is performed on model networks with various structural properties in Sec. III. The dependence of the performance on the structural features is also discussed. In Sec. IV, we reproduce the dismantling performance of the real-world networks by controlling the structures of the model networks and discuss the implications. The final section is devoted to summary and discussion.

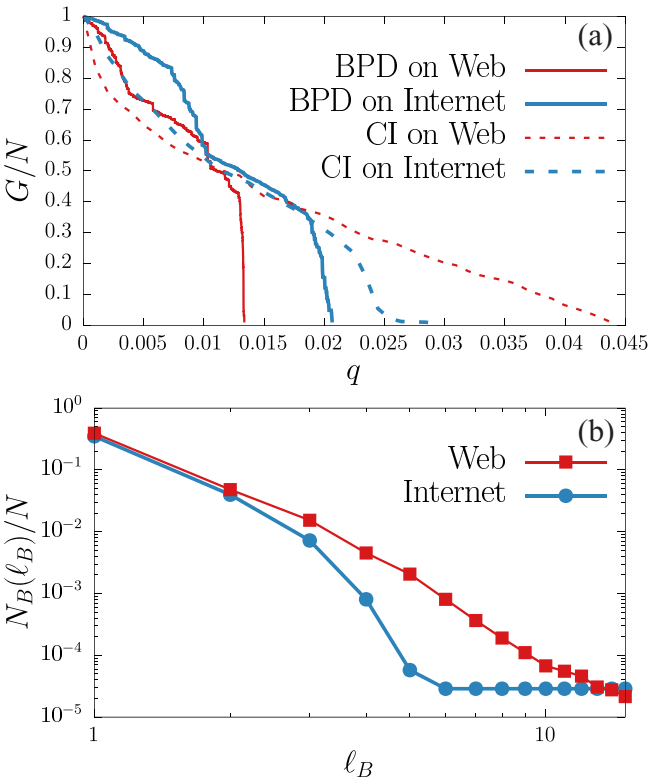

FIG. 1. (a) Performance of the BPD and CI algorithms on two real-world networks. As a fraction q of nodes are deleted, the giant component size is reduced by the fraction G/N, where N is the system size. (b) Fractal scalings of the two real-world networks, the World Wide Web and the Internet at the autonomous level, measured by a random sequential box-covering method [27]. The Web follows a fractal scaling and is regarded as fractal, whereas the Internet is regarded as nonfractal.

II. PERFORMANCE ON REAL-WORLD FRACTAL AND NONFRACTAL NETWORKS

Networks can be factored into a skeleton and shortcuts. The skeleton is a spanning tree formed by the $N-1$ links with the highest betweenness centrality [23,24] or load [25], and shortcuts connect different branches of the tree, forming loops of various sizes. In particular, a skeleton formed by the critical branching process, in which the mean number of offspring is unity, seems to be required for a network to be fractal [21,26]. If the shortcuts are local, then the fractality of the skeleton is preserved, and the resulting network is still fractal. If we add global shortcuts, then they deform the fractal scaling behavior of the skeleton, and the network becomes nonfractal. As the performance of BP-based algorithms relies on the number of local and global loops, it is suggested that the fractality affects the performance of dismantling algorithms.

Figure 1 supports this suggestion, showing that the order of the resilience of sample networks can be changed if one uses different algorithms to obtain the optimal percolation threshold. One of the sample networks is the World Wide Web [28], which is considered to be an undirected network. Two nodes are regarded as connected if there is a hyperlink from one to the other. It is a scale-free network with degree

exponent $\gamma \approx 2.6$, and it appears to be fractal [21]. Another sample is the Internet topology at the autonomous level as collected in early 2010 [29]. It has a power-law degree distribution with degree exponent $\gamma \approx 2.1$, and it is a nonfractal network [21].

Although the BPD algorithm performed better on both networks, the performance gap between the BPD and CI algorithms is small on the Internet, which is nonfractal (Fig. 1). However, the gap is large on the Web, which is fractal. It turns out that the Internet is more vulnerable than the Web to the CI algorithm; however, the Web is more fragile under the BPD algorithm. As these two networks have many differences in features such as system size, degree exponent, and fractality, we need to identify the possible factors that affect the gap between the performance of the two algorithms. We also need to generate model networks with the desired topological characteristics such as system size, degree distribution, and fractality.

III. PERFORMANCE ON FRACTAL AND NONFRACTAL MODEL NETWORKS

The fractal network model (FNM) introduced in Ref. [21] can be used to construct model networks with desired loop characteristics. First, we build a critical branching tree of the desired size with N nodes and $L = N - 1$ links. The degree distribution of the critical tree is controlled by the probability b_m of generating m offspring. For $b_m \sim m^{-\gamma}$ with $\sum_m m b_m = 1$, a critical branching tree with the degree exponent γ is generated. Its fractal dimension is determined as $d_B = (\gamma - 1)/(\gamma - 2)$ [21,26]. Then we add shortcuts to the tree as follows. First, we add stubs to each node of the critical tree, the number of which is proportional to the degree of the node. The total number of stubs is given as $2sL$, where s is a control parameter. Next, we add sL shortcuts between unconnected stubs at different nodes. To maintain the fractal nature, we limit the hopping distance d between nodes that are to be connected by a shortcut. This limitation is required for local loops to conserve the global connectivity and the fractality of the branching tree. A fractal network can be deformed to a nonfractal network by rewiring the fraction r of sL shortcuts without either limiting the distance or changing the degree distribution. By rewiring links, local shortcuts can be changed to long-range loops, which reduces the network diameter or destroys distinct modules. As more shortcuts are rewired, the network loses more of its modularity, and the fractality is broken further, so the network becomes nonfractal.

It is noteworthy that when the degree exponent of the branching tree is close to 2, the network is more centralized at the hub. Then the network becomes nonfractal even if the shortcuts were added locally without the rewiring process [Fig. 2(a)]. This is because the degree of hubs is so large that the network connections are centralized at the hub, and the diameter is easily reduced by connecting neighbors on different hubs within the shortcut distance limitation. On the other hand, as shown in Fig. 2(b), a critical branching tree with a large-degree exponent ($\gamma = 2.7$) maintains its fractality when local shortcuts are added, but it becomes nonfractal when the shortcuts are rewired.

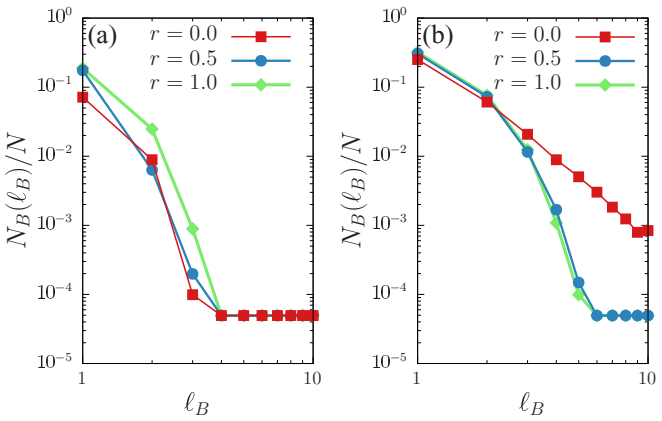

FIG. 2. Fractal scaling behaviors of the fractal model networks measured by the so-called random sequential box-covering method [27]. Networks (a) and (b) are generated on the basis of critical branching trees with degree exponent $\gamma = 2.1$ and $\gamma = 2.7$, respectively. On each skeleton, sL shortcuts are added within the hopping distance $d = 8$, where $s = 3.0$, and $N = L + 1$ is the size of the critical branching tree. The network size was set to $N = 2.0 \times 10^4$. Shortcuts were initially drawn locally ($r = 0.0$), and a fraction r of those shortcuts were rewired randomly ($r = 0.5, 1.0$). For $\gamma = 2.1$, the network is nonfractal even when shortcuts are not rewired. For $\gamma = 2.7$, the network is fractal when shortcuts are added locally and becomes nonfractal as the shortcuts are rewired.

For a given set of parameters such as the degree exponent γ, shortcut parameter s, and rewiring ratio r, we generate 10^3 individual realizations and obtain the performance of the BPD and CI algorithms on these realizations. Because the algorithms were designed to thoroughly break the giant component into subextensive (small) clusters, they provide good estimates of each optimal value around $G = 0$ but can be incorrect far from $G = 0$. We present the distributions of the optimal percolation threshold as the proportion of deleted nodes needed to reduce G to less than 1% of its original size, rather than showing each curve from $G/N = 1$ to $G/N = 0$, which represents the response of the network to the algorithm.

A. Dependence on shortcut rewiring ratio

To determine how the performance of each dismantling algorithm depends on the fractality, we fix the parameters used to construct the networks, such as the degree exponent γ and shortcut parameter s, but control the shortcut rewiring ratio r. Fractal networks are generated using $\gamma = 2.7$ and $r = 0$, whereas nonfractal networks are generated using $\gamma = 2.1$ and arbitrary r values.

We obtain the optimal percolation threshold of each fractal model network using the BPD and CI algorithms. The distributions of those thresholds using each algorithm are obtained from different realizations for a given parameter set of the network structure. As hypothesized, the fractality affects the performance of each algorithm. For the model networks with $\gamma \approx 2.1$ (Fig. 3), the separation between the distributions is unclear; thus, they overlap greatly, even though the BPD algorithm looks slightly better than the CI algorithm for both $r = 0$ and $r = 1$. The ratio of the performance values of the BPD

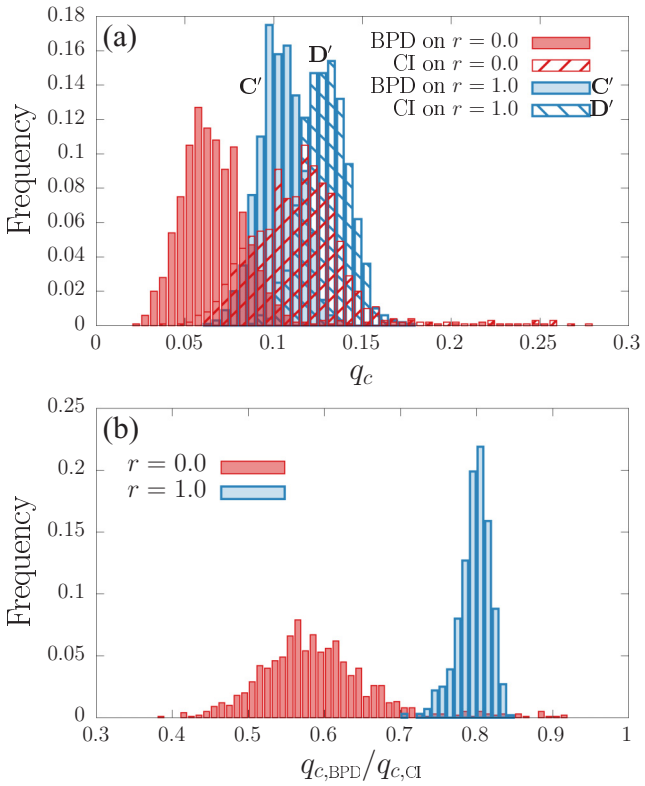

FIG. 3. (a) Performance comparison of the BPD and CI algorithms on scale-free networks generated by the FNM with degree exponent $\gamma \approx 2.1$ for various shortcut rewiring ratios ($r = 0.0, 1.0$). Each bar represents a fraction of q_c, the optimal percolation threshold found by the BPD or CI algorithm. (b) Distribution of the ratio $q_{c,\mathrm{BPD}}/q_{c,\mathrm{CI}}$ of each configuration. The ratio is that of the q_c values found by the BPD and CI algorithms on each graph generated by the FNM. Distributions in both (a) and (b) were obtained from 10^3 realizations. The number of nodes in each model network ranges between 2×10^4 and 2.2×10^4, and the shortcut density is set to $s = 3.0$.

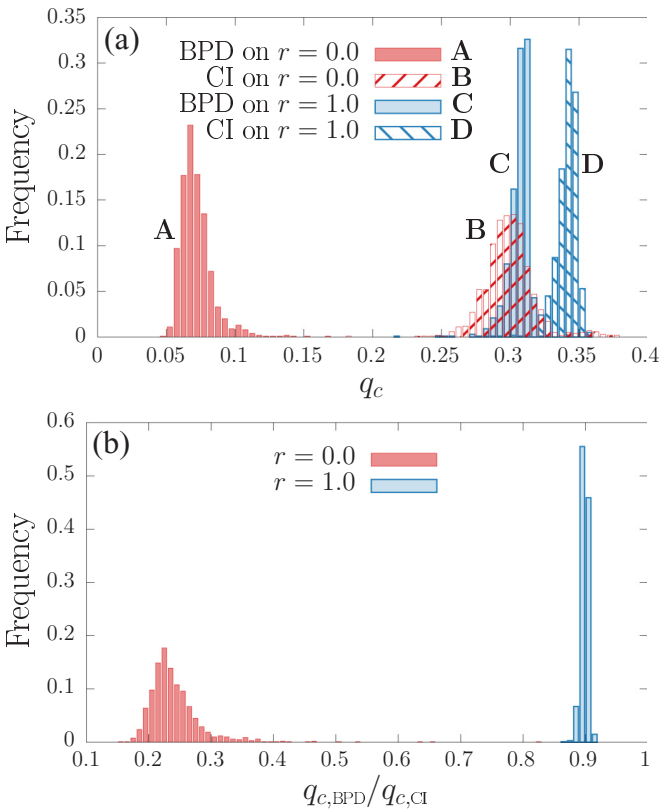

FIG. 4. (a) Performance comparison of the BPD and CI algorithms on scale-free networks generated by the FNM with degree exponent $\gamma \approx 2.7$ and various shortcut rewiring ratios ($r = 0.0, 1.0$). Each bar represents a fraction of q_c, the optimal percolation threshold found by the BPD or CI algorithm, on the ensemble of networks generated by the FNM with the specified shortcut rewiring ratio r. (b) Distribution of the ratio $q_{c,\mathrm{BPD}}/q_{c,\mathrm{CI}}$ of each configuration. The ratio is that of the q_c values found by the BPD and CI algorithms on each graph generated by the FNM. Distributions in both (a) and (b) were obtained from 10^3 realizations. The number of nodes in each model network ranges between 2×10^4 and 2.2×10^4, and the shortcut parameter is set to $s = 3.0$.

and CI algorithms, that is, $q_{c,\mathrm{BPD}}/q_{c,\mathrm{CI}}$, is measured for each realization. Its distribution over different realizations is plotted in Fig. 3(b). Here we notice that $q_{c,\mathrm{CI}}$ is less than $q_{c,\mathrm{BPD}}$ on every single graph instance generated by the FNM. The superiority of BPD over CI appears in every case observed throughout this study. Both algorithms find that the fully rewired ($r = 1$) networks are harder to destroy than the networks with only local shortcuts ($r = 0$).

For the network with $\gamma \approx 2.7$ (Fig. 4), the separation between the distributions of q_c for the BPD and CI algorithms is clear, and those distributions do not overlap greatly. In particular, the difference in performance is much larger for the model networks with $r = 0$ (fractal) than for those with $r = 1$ (nonfractal). As shown in Fig. 4(b), the distribution of the ratio $q_{c,\mathrm{BPD}}/q_{c,\mathrm{CI}}$ over different realizations is located far from unity for $r = 0$ because the BPD and CI algorithms perform very differently on fractal networks. The BPD algorithm is especially effective on fractal networks. However, it is close to unity for $r = 1$, implying that the BPD and CI algorithms exhibit comparable performance on nonfractal networks.

B. Dependence on degree distribution

The performance of dismantling algorithms also depends on the degree distribution of networks. When the shortcut rewiring ratio is fixed at $r = 1$, the model networks are nonfractal for any choice of $2 \leqslant \gamma \leqslant 3$. Thus, on the basis of the previous results, we expect that the distributions of q_c for the BPD and CI algorithms are not widely separated. Figure 5(a) confirms this expectation. For every choice of γ, the q_c distributions for the BPD and CI algorithms are not obviously separated. Figure 5(b) also shows that the performance gap between the BPD and CI algorithms is not large on nonfractal networks with various degree exponents γ. On the other hand, the performance gap between these algorithms varies with the degree exponent γ at $r = 0$ (Fig. 6). When $\gamma = 2.1$, the two distributions overlap to some extent even at $r = 0$. As γ is increased, the separation between the distributions of $q_{c,\mathrm{BPD}}$ and $q_{c,\mathrm{CI}}$ increases. The reason is that a model network possesses more obvious fractality at $r = 0$ for larger γ. As the

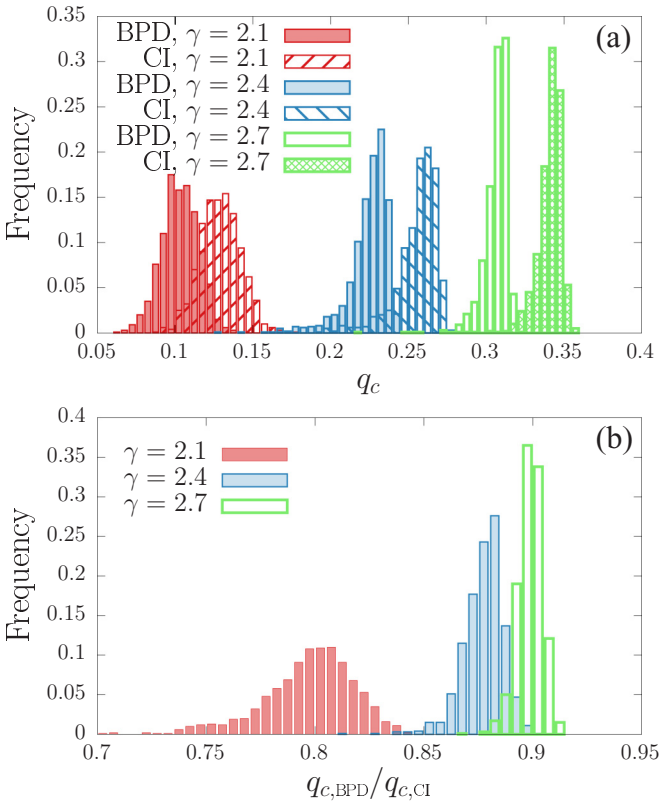

FIG. 5. (a) Performance comparison of the BPD and CI algorithms on scale-free networks generated by the FNM with shortcut rewiring ratio $r = 1.0$ and various degree exponents ($\gamma \approx 2.1$, 2.4, and 2.7). Each bar represents a fraction of q_c, the optimal percolation threshold obtained by the BPD or CI algorithm, on the ensemble of networks generated by the FNM with the specified degree exponent γ. (b) Distribution of $q_{c,\mathrm{BPD}}/q_{c,\mathrm{CI}}$, the ratio of the q_c values obtained by the BPD and CI algorithms, on each realization generated by the FNM. Distributions in both (a) and (b) were obtained from 10^3 realizations. The number of nodes in each model network ranges between 2×10^4 and 2.2×10^4, and the shortcut parameter is set to $s = 3.0$.

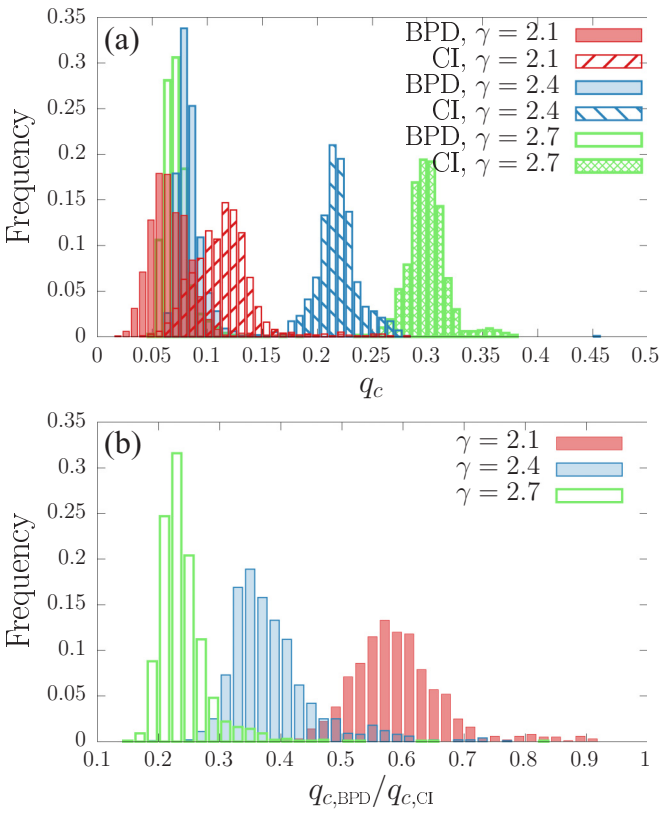

FIG. 6. (a) Performance comparison of the BPD and CI algorithms on scale-free networks generated by the FNM with shortcut rewiring ratio $r = 0.0$ (not rewired) and various degree exponents ($\gamma \approx 2.1$, 2.4, and 2.7). Each bar represents a fraction of q_c, the optimal percolation threshold found by the BPD or CI algorithm, on the ensemble of networks generated by the FNM with the specified degree exponent γ. (b) Distribution of $q_{c,\mathrm{BPD}}/q_{c,\mathrm{CI}}$, the ratio of the values q_c found by the BPD and CI algorithms, on each realization generated by the FNM. Distributions in both (a) and (b) were obtained from 10^3 realizations. The number of nodes in each model network ranges between 2×10^4 and 2.2×10^4, and the shortcut parameter is set to $s = 3.0$.

fractality become more evident, so does the performance gap between the algorithms.

C. Dependence on the system size

Even though the system sizes N are different, the distributions of q_c for each algorithm have similar shapes for nonfractal networks ($r = 1$, $\gamma \approx 2.4$) in Fig. 7. For fractal networks ($r = 0$, $\gamma \approx 2.4$), the BPD algorithm generates a smaller q_c for larger N (Fig. 8). The distribution of q_c for the CI algorithm remains almost unchanged compared to that of the BPD algorithm. As a result, the performance gap becomes larger when the two algorithms are applied on the fractal network of a larger system. Because we know that the FNM generates a network with more manifest fractality for larger N, this is consistent with the previous observation that the BPD algorithm is more effective than the CI algorithm, especially on fractal networks.

IV. REPRODUCTION OF OPTIMAL PERCOLATION OF REAL-WORLD NETWORKS FROM MODEL NETWORKS

In Sec. II, we observed a nontrivial phenomenon: The BPD algorithm reveals that the Web is more vulnerable to attack than the Internet, whereas the CI algorithm shows the opposite [Fig. 1(a)]. Here we show that this phenomenon is not a coincidence occurring on the particular pair of real-world networks but rather a generic phenomenon. That is, this phenomenon can be reproduced by any pair of networks categorized as fractal and nonfractal networks with the same degree distributions.

We first remark that the BPD and CI algorithms perform very differently on fractal networks. This can be recognized in Fig. 4(a) for the fractal case with $r = 0$ and $\gamma = 2.7$, in which distributions A and B are widely separated. This large separation can also be found in the Web among real-world networks. Recall that the Web is a fractal network with the degree exponent $\gamma \approx 2.6$.

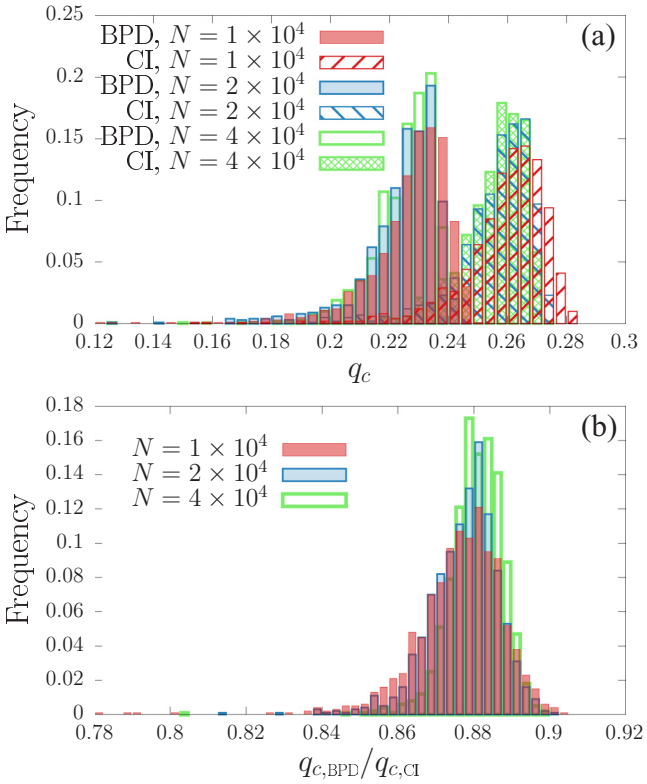

FIG. 7. (a) Performance comparison of the BPD and CI algorithms on scale-free networks generated by the FNM with shortcut rewiring ratio $r = 1.0$ and degree exponent $\gamma \approx 2.4$ for various total numbers of nodes ($N = 1.0 \times 10^4$, 2.0×10^4, and 4.0×10^4). Each bar represents a fraction of q_c, the optimal percolation threshold found by the BPD or CI algorithm, on the ensemble of networks generated by the FNM with the specified total node number N. (b) Distribution of $q_{c,\mathrm{BPD}}/q_{c,\mathrm{CI}}$, the ratio of the q_c values found by the BPD and CI algorithms, on each realization generated by the FNM. Distributions in both (a) and (b) were obtained from 10^3 realizations. Shortcut parameter is set to $s = 3.0$.

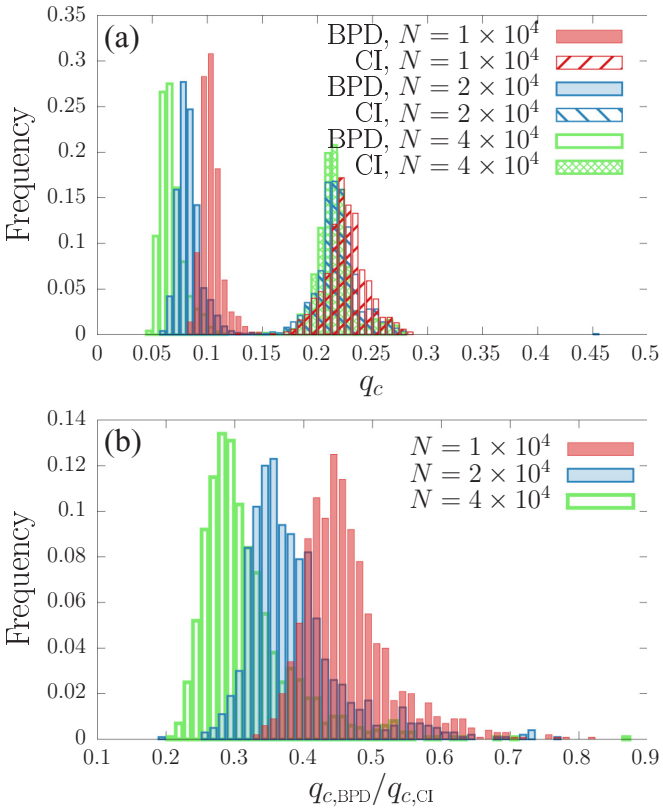

FIG. 8. (a) Performance comparison of the BPD and CI algorithms on scale-free networks generated by the FNM with shortcut rewiring ratio $r = 0.0$ (not rewired) and degree exponent $\gamma \approx 2.4$ for various total numbers of nodes ($N = 1.0 \times 10^4$, 2.0×10^4, and 4.0×10^4). Each bar represents a fraction of q_c, the optimal percolation threshold found by the BPD or CI algorithm, on the ensemble of networks generated by the FNM with the specified total node number N. (b) Distribution of $q_{c,\mathrm{BPD}}/q_{c,\mathrm{CI}}$, the ratio of the q_c values found by the BPD and CI algorithms, on each realization generated by the FNM. Distributions in both (a) and (b) were obtained from 10^3 realizations. Shortcut parameter is set to $s = 3.0$.

Next, as we found in the previous section, the BPD and CI algorithms show comparable performance on nonfractal networks. The narrow separation between distributions C and D in Fig. 4(a) for a nonfractal model network with $r = 1.0$ and $\gamma = 2.7$ represents similar performance. However, distributions C and D are not sufficient to reflect the performance of the algorithms on the Internet, because the positions of $q_{c,\mathrm{BPD}}$ and $q_{c,\mathrm{CI}}$ on the Internet are located between those on the Web [Fig. 1(a)], whereas the distributions of C and D are located outside of the region of distribution B.

To reproduce the ordering of q_c values obtained in Fig. 1(a), we need to shift distributions C and D into the region between distributions A and B. Recall that distributions C and D were obtained on nonfractal model networks with $\gamma \approx 2.7$, whereas the Internet has $\gamma \approx 2.1$. On the basis of the previous result, we may guess that reducing the degree exponent γ will move distributions C and D to the desired positions. As shown in Fig. 5(a), the distributions of the CI and BPD algorithms on nonfractal networks move to the left as the degree exponent decreases. On the basis of this result, we set a smaller degree exponent ($\gamma \approx 2.1$) for the nonfractal network. Figure 9 is obtained using this setting. Distributions C′ and D′ on nonfractal networks are located between distributions A and B on fractal networks, successfully reproducing the desired order of q_c values on the Internet and the Web.

Thus, we conclude that if we have a pair of scale-free networks—one a fractal network with sufficiently large degree exponent γ and the other a nonfractal network with small γ—and we dismantle them using the BPD and CI algorithms, it is highly probable that the fractal network is more vulnerable than the nonfractal network to attack by the BPD algorithm. Conversely, the nonfractal network is more vulnerable than the fractal network when the CI algorithm is used.

V. CONCLUSION AND DISCUSSION

We studied the dismantling behavior of two well-known heuristic algorithms for the optimal percolation problem, the CI and BPD algorithms. Both algorithms were developed assuming that the target network is locally treelike. However, many real-world networks contain short-range and long-range

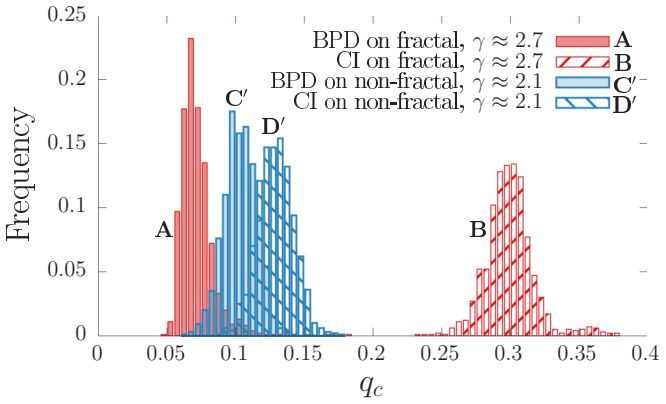

FIG. 9. Distributions of $q_{c,\mathrm{BPD}}$ and $q_{c,\mathrm{CI}}$ on fractal networks of degree exponent $\gamma \approx 2.7$ and nonfractal networks of degree exponent $\gamma \approx 2.1$. For both ensembles of fractal and nonfractal networks, 10^3 realizations were generated by the FNM with the shortcut parameter $s = 3.0$ and rewiring ratio $r = 1.0$. Note that distributions A and B are the same as A and B in Fig. 4, and C′ and D′ are the same as C′ and D′ in Fig. 3.

loops. It is interesting to consider how the performance of those algorithms depends on the loop structure. Here we constructed an FNM in which the loop structure is contained and controlled. FNMs were generated using a critical branching tree to which local shortcuts were added. Then those local shortcuts were rewired randomly according to a shortcut rewiring parameter r. When $r = 0$, the network contains only local shortcuts and becomes fractal when the degree exponent is sufficiently large. When $r = 1$, every shortcut is rewired randomly. The network becomes nonfractal regardless of the degree exponent.

Overall, the BPD algorithm exhibited better performance for both fractal and nonfractal networks. The performance of the BPD algorithm was much better than that of the CI algorithm on fractal networks; however, the algorithms are comparable on nonfractal networks. Fractal networks contain local community structures that are linked by sparse weak ties. Intuitively, those communities can be easily separated when the nodes on those weak ties are removed. The BPD algorithm recognizes such target nodes efficiently; however, the CI algorithm is less efficient in finding them (Fig. 4). On the other hand, nonfractal networks are, in general, globally entangled, and the two algorithms show similar performance.

The difference and similarity of the dismantling efficiency on fractal and nonfractal networks, respectively, can also be found in a pair of real-world networks, the Web and the Internet (Fig. 1). They were regarded as a fractal and a nonfractal network, respectively [21,22]. We revealed that for the Web, there is a large difference between $q_{c,\mathrm{BPD}}$ and $q_{c,\mathrm{CI}}$, whereas for the Internet they are similar. They are ordered as follows: Whereas the $q_{c,\mathrm{BPD}}$ value of the Web is smaller than that of the Internet, the $q_{c,\mathrm{CI}}$ value of the Web is larger than that of the Internet, as shown in Fig. 1. This implies that although the Web is more vulnerable than the Internet to the BPD algorithm, the Internet is more vulnerable to the CI algorithm. To explain this phenomenon, we notice that the Web and the Internet have different degree exponents, $\gamma \approx 2.6$ and $\gamma \approx 2.1$, respectively. Because of the smaller degree exponent, the hub of the Internet has a larger degree, making the network more fragile. However, this fragility is not as great as that of the Web, which originates from the fractality. Thus, the $q_{c,\mathrm{BPD}}$ value of the Web is smaller than any other q_c value. We remark that $q_{c,\mathrm{BPD}}$ and $q_{c,\mathrm{CI}}$ on the Internet are smaller than $q_{c,\mathrm{CI}}$ on the Web. This phenomenon was reproduced on a pair of model networks with fractality and degree exponents similar to the Web and the Internet (Fig. 9).

Thus far, we have investigated the performance of fractal model networks with a fixed shortcut parameter, $s = 3$. Here we consider how the performance depends on the parameter s. We first consider the case when $r = 0$, that is, there is no link-rewiring process. As s is increased, the network becomes more locally entangled. Then the difference between the performance of the BPD and CI algorithms becomes larger, because the CI algorithm does not work well on more entangled networks. However, when r is large, the network becomes less locally entangled by link rewirings, and the network becomes nonfractal. Thus, the performance gap is small. As s is increased, the gap becomes even smaller.

Finally, we summarize the dependence of the dismantling performance on the structural properties of the fractal model networks as a function of the system size N, degree exponent γ, shortcut parameter s, and link-rewiring ratio r in Table I.

TABLE I. Dependence of the performance values $q_{c,\mathrm{BPD}}$ and $q_{c,\mathrm{CI}}$ and their difference, $\Delta \equiv q_{c,\mathrm{CI}} - q_{c,\mathrm{BPD}}$, on the system size N, degree exponent γ, shortcut parameter s, and rewiring ratio r of the fractal model network in the leftmost column. ↑ (↓, ∅) represents increasing (decreasing, insensitive) behavior of each quantity in the top row as the corresponding parameter in the leftmost column is increased, where the other parameters are fixed.

	$q_{c,\mathrm{BPD}}$	$q_{c,\mathrm{CI}}$	$\Delta = q_{c,\mathrm{CI}} - q_{c,\mathrm{BPD}}$
N	↓	↓	↑ $(r = 0)$/ ∅$(r = 1)$
γ	↑	↑	↑ $(r = 0)$/ ↓ $(r = 1)$
s	↑	↑	↑ $(r = 0)$/ ↓ $(r = 1)$
r	↑	↑	↓

ACKNOWLEDGMENTS

This work was supported by the National Research Foundation of Korea by Grant No. NRF-2014R1A3A2069005.

APPENDIX A: COLLECTIVE INFLUENCE ALGORITHM

Although the CI algorithm [9] starts theoretically from sophisticated considerations of local stability analysis of message-passing equations, it provides a simple centrality measure as a criterion for selection of nodes to be deleted. Assuming that the network is locally treelike, the solution with a vanishing giant connected component depends on the largest eigenvalue of the modified nonbacktracking matrix. The solution is stable only when the eigenvalue is less than unity, whereas the eigenvalue drops abruptly from one to zero when no loop remains. By using a perturbative method, the problem is reduced to minimizing the cost function, which is defined as

the sum of $\mathcal{C}_{\ell}^{(\mathrm{CI})}(i)$ over all nodes, where

$$\mathcal{C}_{\ell}^{(\mathrm{CI})}(i) = (k_i - 1) \sum_{j \in \partial \mathrm{Ball}(i,\ell)} (k_j - 1).$$

Here $\partial \mathrm{Ball}(i, \ell)$ represents the nodes on the surface of a ball centered at node i with radius ℓ. The CI algorithm repeatedly removes the node with the largest CI value until the largest connected component vanishes. The CI value of every node is reevaluated after each removal. The algorithm outperforms intuitive decimations based on traditional centrality measures such as the degree or eigenvalue centrality, because it can take into account the importance of *weak nodes* with small degrees. Although the algorithm becomes exact as $\ell \to \infty$ for an infinite treelike network, a small ℓ still yields good estimation for finite networks. Moreover, deleting a fixed fraction of nodes with the largest CI values at once does not affect the performance in typical cases, allowing the algorithm to work in a time complexity of $O(N \log N)$. In this study, the CI algorithm uses $\mathcal{C}_{\ell=2}^{(\mathrm{CI})}(i)$ as its criterion because this value is more effective than larger or smaller ℓ for the prototypical system size of the model networks used here. When we take an excessively large ℓ value, the computation time becomes long and the performance is degraded, because the algorithm performs only random deletion when ℓ is equal to or larger than the network diameter. A 0.1% portion of the nodes of the original network were deleted at each step until the size G of the giant component reached 1% of the number of nodes of the original network.

APPENDIX B: BELIEF-PROPAGATION-GUIDED DECIMATION ALGORITHM

The BPD algorithm [17] uses the minimum feedback vertex set (mFVS) problem as an approach to the optimal percolation problem. The mFVS problem is to find a minimal set of nodes whose removal eliminates every loop. If we draw a subgraph on the original graph by retaining only nodes that have one parent, the subgraph consists only of simple loops and trees [30]. This rule is local, so it can be expressed by BP equations involving the variables of neighboring nodes. The algorithm evaluates the marginal empty probability q_0^i of each node at each moment from the probabilities in the cavity graphs, which are calculated by iteration of the BP equations. Definitions of q_0^i and the BP equations can be found in Ref. [17]. The node with the highest q_0^i is removed because removal of that node is strongly recommended in order to draw a subgraph without loops. Although the BP equations are not guaranteed to converge to a fixed point on general graphs, the equations are iterated a fixed number of times in this algorithm. In practice, multiple nodes with the highest q_0^i are deleted together in one step. After the evaluation of q_0^i and node removal, another cycle of iteration and node removal is repeated until no loop remains.

The resultant tree components are broken into pieces by removing additional nodes until no remaining connected component is larger than expected. Because our purpose is to decompose the giant component, small components with loops can be allowed. Thus, among the deleted nodes, some nodes are revived unless a large component emerges upon their revival. The BPD algorithm reportedly outperforms the CI algorithm on various types of models and real-world networks [17]. The BPD algorithm is based on the spin glass model, where each possible microscopic state can be realized with a probability weighted by the number of remaining nodes multiplied by the reweighting parameter X. In this study, the reweighting parameter X is set to 12.0. When it is sufficiently large, it does not affect the performance of the algorithm significantly, even though the shape of the curve for each trial can vary slightly. At each step, the BP equations are iterated, and 1% of the remaining nodes are deleted. Then, the remaining tree components are broken into pieces by deleting additional nodes until the size G of the giant component reaches 1% of the number of nodes in the original network.

[1] J.-G. Liu, J.-H. Guo, Q. Guo, and T. Zhou, Sci. Rep. **6**, 21380 (2016).

[2] D.-B. Chen, H. Gao, L. Lü, and T. Zhou, PLoS ONE **8**, e77455 (2013).

[3] D. Chen, L. Lü, M.-S. Shang, Y.-C. Zhang, and T. Zhou, Physica A (Amsterdam) **391**, 1777 (2012).

[4] S. Yeruva, T. Devi, and Y. S. Reddy, Physica A (Amsterdam) **452**, 133 (2016).

[5] Y. Liu, B. Wei, Y. Du, F. Xiao, and Y. Deng, Chaos Solitons Fractals **86**, 1 (2016).

[6] M. Kitsak, L. K. Gallos, S. Havlin, F. Liljeros, L. Muchnik, H. E. Stanley, and H. A. Makse, Nat. Phys. **6**, 888 (2010).

[7] M. Richardson and P. Domingos, in *Proceedings of the 8th ACM SICKDD International Conference on Knowledge Discovery and Data Mining* (ACM, New York, NY, 2002), pp. 61–70.

[8] A. Y. Lokhov and D. Saad, Proc. Natl. Acad. Sci. U.S.A. **114**, E8138 (2017).

[9] F. Morone and H. A. Makse, Nature **524**, 65 (2015).

[10] P. Clusella, P. Grassberger, F. J. Pérez-Reche, and A. Politi, Phys. Rev. Lett. **117**, 208301 (2016).

[11] D. Kempe, J. Kleinberg, and É. Tardos, in *Proceedings of the 9th ACM SICKDD International Conference on Knowledge Discovery and Data Mining* (ACM, New York, NY, 2003), pp. 137–146.

[12] R. M. Karp, in *Complexity of Computer Computations* (Springer, Berlin, 1972), 85–103.

[13] L. Lü, D. Chen, X.-L. Ren, Q.-M. Zhang, Y.-C. Zhang, and T. Zhou, Phys. Rep. **650**, 1 (2016).

[14] P. Holme, B. J. Kim, C. N. Yoon, and S. K. Han, Phys. Rev. E **65**, 056109 (2002).

[15] F. Altarelli, A. Braunstein, L. Dall'Asta, J. R. Wakeling, and R. Zecchina, Phys. Rev. X **4**, 021024 (2014).

[16] J. S. Yedidia, W. T. Freeman, and Y. Weiss, IEEE Trans. Inf. Theory **51**, 2282 (2005).

[17] S. Mugisha and H.-J. Zhou, Phys. Rev. E **94**, 012305 (2016).

[18] A. Braunstein, L. Dall'Asta, G. Sermerjian, and L. Zdeborová, Proc. Natl. Acad. Sci. U.S.A. **113**, 12368 (2016).

[19] M. Mézard and A. Montanari, in *Information, Physics and Computation* (Oxford University Press, Oxford, 2009), 310–315.
[20] B. Karrer, M. E. J. Newman, and L. Zdeborova, Phys. Rev. Lett. **113**, 208702 (2014).
[21] K.-I. Goh, G. Salvi, B. Kahng, and D. Kim, Phys. Rev. Lett. **96**, 018701 (2006).
[22] C. Song, S. Havlin, and H. A. Makse, Nature (London) **433**, 392 (2005).
[23] L. C. Freeman, Sociometry **40**, 35 (1977).
[24] M. Girvan and M. E. J. Newman, Proc. Natl. Acad. Sci. U.S.A. **99**, 7821 (2002).
[25] K.-I. Goh, B. Kahng, and D. Kim, Phys. Rev. Lett. **87**, 278701 (2001).
[26] Z. Burda, J. D. Correia, and A. Krzywicki, Phys. Rev. E **64**, 046118 (2001).
[27] J. S. Kim, K.-I. Goh, B. Kahng, and D. Kim, Chaos **17**, 026116 (2007).
[28] R. Albert, H. Jeong, and A.-L. Barabási, Nature (London) **401**, 130 (1999).
[29] University of Oregon Route Views Archive Project, http://archive.routeviews.org/.
[30] H.-J. Zhou, Eur. Phys. J. B **86**, 455 (2013).