RESEARCH Open Access

Revisiting the power of reinsertion for optimal targets of network attack

Changjun Fan[1*†], Li Zeng[1†], Yanghe Feng[1], Baoxin Xiu[2], Jincai Huang[1] and Zhong Liu[1]

Abstract
Understanding and improving the robustness of networks has significant applications in various areas, such as bioinformatics, transportation, critical infrastructures, and social networks. Recently, there has been a large amount of work on network dismantling, which focuses on removing an optimal set of nodes to break the network into small components with sub-extensive sizes. However, in our experiments, we found these state-of-the-art methods, although seemingly different, utilize the same refinement technique, namely reinsertion, to improve the performance. Despite being mentioned with understatement, the technique essentially plays the key role in the final performance. Without reinsertion, the current best method would deteriorate worse than the simplest heuristic ones; while with reinsertion, even the random removal strategy achieves on par with the best results. As a consequence, we, for the first time, systematically revisit the power of reinsertion in network dismantling problems. We re-implemented and compared 10 heuristic and approximate competing methods on both synthetic networks generated by four classical network models, and 18 real-world networks which cover seven different domains with varying scales. The comprehensive ablation results show that: i) HBA (High Betweenness Adaption, no reinsertion) is the most effective network dismantling strategy, however, it can only be applicable in small scale networks; ii) HDA (High Degree Adaption, with reinsertion) achieves the best balance between effectiveness and efficiency; iii) The reinsertion techniques help improve the performance for most current methods; iv) The one, which adds back the node based on that it joins the clusters minimizing the multiply of both numbers and sizes, is the most effective reinsertion strategy for most methods. Our results can be a survey reference to help further understand the current methods and thereafter design the better ones.

Keywords: Network dismantling, Reinsertion

Introduction

Many real-world systems can be described through the complex network perspective, including air transport [17], power grid [3], malicious organization [9, 10], Internet [3] or inter-personal networks [15]. One of the most important topics on these networks is about the robustness, i.e., the capacity to maintain the functionality after a major failure [29]. Since connectivity is the fundamental basic for almost all behaviors on networks, researches thus try to quantify how the connectivity is affected by node(or link) removal, and there comes with the well-defined network dismantling problem [1], which aims at identifying an optimal sequence of nodes that maximizes the damage on the network connectivity [5]. Such analysis yields a wide range of practical applications, such as immunize the epidemic propagation in populations [23], block the rumor spreading on social networks [15], prevent the virus diffusion in computer networks [7], etc.

However, the exact solution is computationally intractable for medium and large networks due to its NP-hard nature [5], thus a large number of approximate methods have been proposed, including the heuristic methods [4, 11, 12, 21, 23, 31], and some message-passing algorithms [5, 22]. The former methods often greedily

*Correspondence: fanchangjun09@163.com
†Changjun Fan and Li Zeng contributed equally to this work.
[1]College of Systems Engineering, National University of Defense Technology, Changsha, Hunan, China
Full list of author information is available at the end of the article

select target nodes based on local metrics, like node degree, which often leads to sub-optimal solutions; the latter ones are more accurate and global, while they need to iterate certain steps on the whole network to select the suitable candidate nodes [31], which would sacrifice some efficiency.

Although these methods looks different from each other, many of them [5, 21, 22, 24, 31] share the same refinement technique, named reinsertion (we later introduce it in detail in Section 2), which is just simply mentioned in the respective literature, while has significant influence on the final results. As illustrated in Fig. 1, we draw the robustness curves ("Robustness measure" section) of random removal, simplest heuristic HDA and the representative CI (details of these methods will be introduced in "Competing methods" section) on a real-world Gnutella31 network [18]. We can see that without reinsertion, the representative method CI cannot even beat the simplest heuristic HDA, while with reinsertion, the random removal strategy can achieve comparable performance than the state-of-the-art results. In some literature, people just compare their methods enhanced with reinsertion with others without reinsertion, and then report the 'fake' superiority of their model, since we are not sure whether the superiority comes from the model itself or just the reinsertion. Such confused results prevent us from selecting the best algorithm to handle the application at hand.

In this paper, we systematically investigate the power of reinsertion on the current methods for network dismantling. As far as we know, there are no previous efforts that conduct such comprehensive ablation studies for the reinsertion. We aim at figuring out the following three questions: i) *Which is the current best method if all without reinsertion?* ii) *Which one is the best if all with reinsertion?* iii) *Which is the best reinsertion strategy?*

To achieve this, we conduct ablation study (with/without reinsertion) for all the current network dismantling methods, including both traditional heuristics and the state-of-art message-passing ones, on synthetic networks and real-world networks. We use four random network models, including ER [8], WS [30], BA [3] and PLC [13], to generate diverse graphs with varying sizes and structures by controlling the model parameters. For real-world networks, we select 18 real networks covering 7 domains and with different scales. Considering that the network robustness can be described by different measures, we choose the area under the robustness curve as the main evaluation metric, since it captures the response of the whole dismantling process. Extensive experiments demonstrate that the reinsertion can significantly improve the performance regardless of the network types and the methods. Besides, since reinsertion is rather effective for the network dismantling problem, perhaps people should focus on this technique itself rather than other aspects, so as to design a better attack strategy.

The main contributions of this paper are summarized as follows:

1. We conduct comprehensive ablation studies that are with and without reinsertion for the network dismantling problem. We compare 10 competing

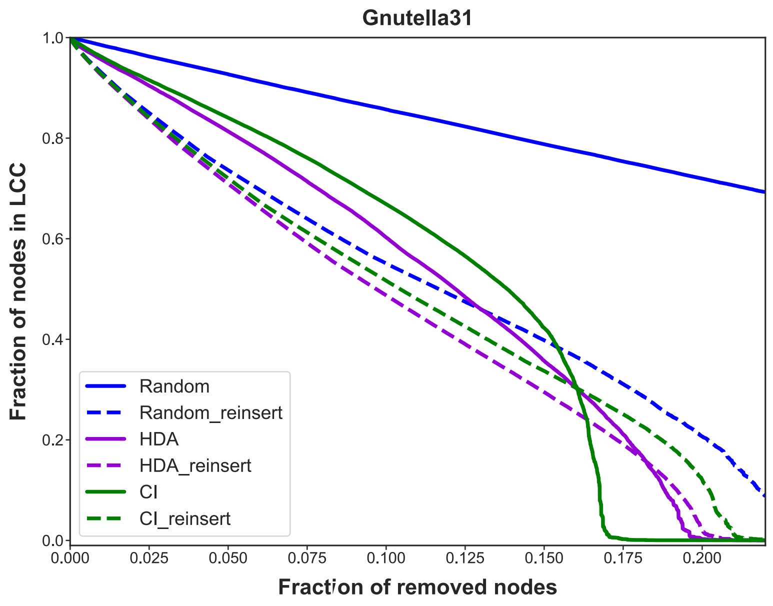

Fig. 1 Illustration of the reinsertion power on Gnutella31 network

methods on both synthetic graphs generated from four random network types and 18 real-world networks covering seven domains and scales up to hundreds of thousands nodes;
2. We design two other reinsertion strategies, and empirically prove that they have surpassed the previous reinsertion technique in a large margin;
3. The results obtained in this paper could provide a valuable guide for selecting and designing the most appropriate method for practical network dismantling problems.

The rest of the paper is organized as follows. We introduce the reinsertion method, robustness measures, competing methods and experimental data in "Method" section. We analyze the comprehensive ablation results and effects of different reinsertion strategies in "Results" section. Finally, we conclude the paper in Section 5.

Method

In this section, we introduce the experimental setups. We first introduce the robustness measure to evaluate the dismantling efficacy, then we introduce the reinsertion technique that is widely adopted in most current competitors. After that, we describe the competitors we are to analyze and the experimental data, including both synthetic graphs and real-world networks.

Robustness measure

Network dismantling is to identify a sequence of nodes of which removal would degrade the network connectivity maximally, and this connectivity disintegration is often measured as the relative reduction in the size of the giant(largest) connected component (GCC size) [5, 21]. The smaller the remaining GCC size, the more the network is considered to have been disintegrated.

We consider the area under the robustness curve as the evaluation metric, which is plotted with horizontal axis being the fraction of nodes removed, and the vertical axis being the remaining GCC size. It is defined as:

$$R = \frac{1}{N}\sum_{Q=1}^{N} s(Q) \tag{1}$$

where N is the number of graph nodes, $s(Q)$ is the remaining GCC size after removing Q nodes. Intuitively this measure is equivalent to assessing how many nodes the GCC contains when a new node is deleted from the network, and sum this for all nodes [29]. Note that Eq. 1 captures the network's response to the dismantling throughout the whole process, and the computation of R requires a ranking of the nodes, we are interested in minimizing R over all possible node orders.

In this paper, we evaluate the ablation performance of reinsertion for this robustness measure.

Reinsertion technique

The reinsertion is firstly proposed as an independent strategy for network destruction and immunization [27], and later developed as an important refinement technique for other dismantling strategies. The reinsertion starts from the point, where the network has been dismantled over by a certain strategy, it adds back one of the removed node, chosen such that, once reinserted, it joins the smallest number of clusters. When the node is reinserted, restore the edges with its neighbors which are in the network (but not the ones with neighbors not yet reinserted, if any). Repeat the above the procedure until all the nodes are back in the network.

As is shown in Fig. 2, each node is assigned an index $c(i)$ given by the number of clusters it would join if it is reinserted in the network. The red node has $c(red) = 2$, while the blue one has $c(blue) = 4$, the green node has $c(green) = 3$. Then the node with the smallest $c(i)$ is reinserted, i.e., the red node. After that, the $c(i)$s are recalculated and the new node with smallest $c(i)$ is found and reinserted. Repeat these steps until the terminal criteria meets.

We will later show with extensive experiments how powerful such a simple technique is to the current network dismantling methods.

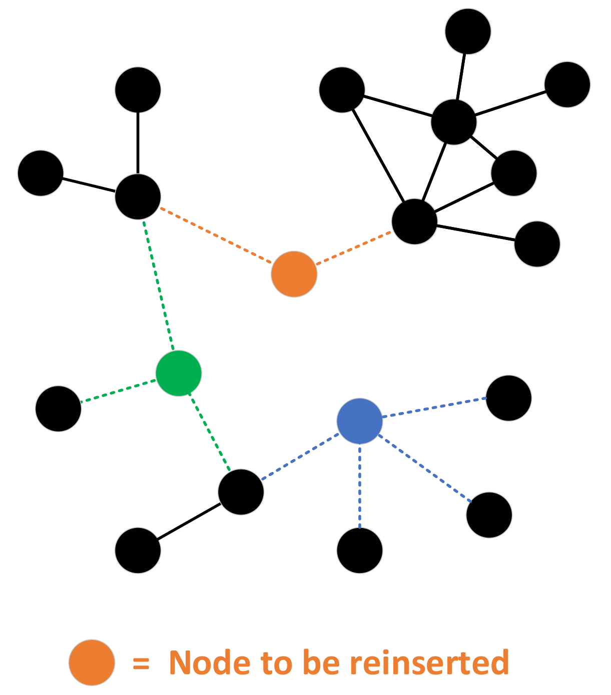

Fig. 2 Illustration of the reinsertion procedure

Competing methods

In this paper, we compare with 9 most representative competing methods. The first five are traditional heuristics which are based on some local or global structure centrality, such as degree, betweenness, closeness, pagerank, or collective influence. The remaining five are specifically designed for dismantling networks. Note that we also add a **Random** removal strategy as a worst possible baseline.

High Degree Adaptive (HDA) [23]. HDA is an adaptive version of high degree method [2]. Within each step, the node with the highest degree is removed, and then the remaining degrees are updated.

High Betweenness Adaptive (HBA) [12]. HBA is the adaptive version of the high betweenness method, where the betweenness centrality of the remaining nodes is recomputed after each node removal. Betweenness centrality of a node equals to the sum of the fraction of all pairs shortest paths that pass through this node. It is a very useful centrality measure that benefits many network-related applications such as community detection and network vulnerability. However, the high computing cost prohibits its use in large-scale problem settings.

High Closeness Adaptive(HCA) [4]. HCA is the adaptive version of the high closeness method. Closeness centrality describes how close a node is to all the other nodes in the graph. It is calculated as the reciprocal of average distances from one node to all the others. Similar as HBA, the high complexity cost prevents its application in large networks.

High PageRank Adaptive (HPRA) [6]. HPRA is the adaptive version of high PageRank method. PageRank has been widely employed in search engines, as it provides a global ranking of all web pages, regardless of their content, based solely on their location in the Web's graph structure [6]. PageRank computes the probabilities for a random-walking agent to reach every node in the network, which is also regarded as useful indications to supervise the network attack.

Collective Influence(CI) [21]. The Collective Influence measure is defined as the product of the node's reduced degree (i.e. original degree minus one) with the sum of the reduced degrees of the nodes that are within a constant hops away from it. This measure describes the proportion of other nodes that can be reached from a given node, assuming the nodes with higher CI values play more crucial roles in networks. The CI method sequentially removes the node with the highest CI value and recalculating the collective influence for the rest following operations.

MinSum [5]. MinSum is proposed to address the network dismantling problem. It consists three stages, which firstly utilizes a variant of message-passing algorithm to break all the cycles, and then breaks the remaining tree into small components by removing a fraction of nodes that vanishes in the large size limit. In the third stage, it greedily reinserts some nodes that close cycles without increasing too much the size the largest component, to reduce the total number of nodes removed.

Belief Propagation-guided Decimation (BPD) [22]. BPD is very similar as MinSum, which contains the same three stages. The difference lies on that BPD treats the decycling problem as the minimum-FVS construction. The FVS refers to the feedback vertex set, which is a set of node that will cause the network to become a forest if being deleted. To solve this problem, BPD proposes a belief propagation-guided decimation algorithm. After, it conducts the same subsequent steps, including tree breaking and node reinsertion.

CoreHD [31]. CoreHD also contains the similar three stages. The only difference lies in the decycling stage. Unlike the message-passing or belief-propagation algorithm, CoreHD instead seeks to remove the minimum nodes to empty the 2-core subgraph in the network, since the network is acyclic equals to that the 2-core subgraph is empty. CoreHD greedily remove the highest degree node in the 2-core subgraph until the end.

GND [25]. GND is the state-of-the-art method to address the network dismantling problem with non-unit removal costs. It first defines a node weighted Laplacian, and then proposes a simple and elegant approximate algorithm to calculate its second smallest eigenvector, based on which the set of nodes are removed. GND repeats the process until the end. Note that the unit-cost GND is just the spectral cut method.

We use SNAP software[1] to implement the heuristic methods, including Random, HDA, HBA, HCA and HPRA. For the other baselines, we use the source codes[2][3][4][5][6] released online, and use the defaut parameter settings for each method.

Synthetic graphs

We evaluate all competitors against various synthetic networks. Synthetic networks are the result of applying generative function, present the advantage of displaying specific topological features that are both a prior known and tunable [29]. More specifically, we select a collection of 4 most common network types, summarized in Table 1. Note that there are many other random network models, such as regular graphs, circle graphs, grid graphs, ladder graphs, etc, we do not consider them since they are not difficult to dismantle, and there always exists some effective heuristic methods for them.

[1]http://snap.stanford.edu/snap/
[2]https://github.com/zhfkt/ComplexCi
[3]http://power.itp.ac.cn/~zhouhj/codes.html
[4]https://github.com/abraunst/decycler
[5]https://github.com/hcmidt/corehd
[6]https://github.com/renxiaolong/Generalized-Network-Dismantling

Table 1 Overview of four random network types

Abbre	Name	Parameters
ER	Erdos-Renyi	n in [500, 800], p in [0.10,0.15,0.18,0.20,0.25],
WS	Watt-Strogatz	n in [500, 800], k in [5,6,7,8,9], p in [0.1,0.2,0.3,0.4,0.5,0.6,0.7]
BA	Barabasi-Albert	n in [500, 800], m in [2,3,4,5,6]
PLC	Powerlaw-Cluster	n in [500, 800], m in [2,3,4,5,6], p in [0.10,0.15,0.18,0.20,0.25]

Erdos-Renyi(ER) [8]. ER model is first introduced by Paul Erdos and Alfred Renyi, it returns a $G_{n,p}$ graph, where n is the graph nodes, p is the edge creation probability. The $G_{n,p}$ chooses each of the possible edges with probability p. This model can be used in the probabilistic method to prove the existence of graphs satisfying various properties, or to provide a rigorous definition of what it means for a property to hold for almost all graphs [8].

Watt-Strogatz(WS) [30]. WS is a random generative model that produces graphs with small-world properties, including short average path lengths and high clustering. It was proposed by Duncan J. Watts and Steven Strogatz in 1998. The tunable parameters include the node number n, k nearest neighbors in a ring topology that each node is joined with, and the probability of rewiring each edge p.

Barabasi-Albert(BA) [3]. BA is a model that generates random scale-free networks using a preferential attachment mechanism. Many real-world networks are thought to be approximately scale-free and contain few nodes (called hubs) with unusually high degree as compared to the other nodes. The BA model tries to explain the existence of such nodes in real networks. The algorithm is named for its inventors Albert-Laszlo Barabasi and Reka Albert and is a special case of a more general model called Price's model [28]. It generates a graph of n nodes by attaching new nodes with each adding m edges that are preferentially attached to existing nodes with high degree.

Powerlaw-Cluster(PLC) [13]. PLC is a mode for generating graphs with powerlaw degree distribution and approximate average clustering. It is essentially the BA growth model with an extra step that each random edge is followed by a chance of making an edge to one of its neighbors too (and thus a triangle) [13]. The model improves on BA in the sense that it enables a higher average clustering to be attained if desired. The tunable parameters include the number of nodes n, the number of random edges to add for each new node m, and the probability of adding a triangle after adding a random edge p.

Figure 3 visualizes one instance for each of the above four networks types.

Real-world networks

We also conduct experiments on 18 real-world networks, which cover a wide range of domains, including malicious networks, PPI networks, infrastructure networks, social networks, citation networks, communication networks, etc. Specifically, they are:

Corruption [26], a malicious network where nodes are people listed in scandals, and the ties indicate that two people were involved in the same corruption scandal;

Crime [16], a malicious network from the projection of a bipartite network of persons and crimes, each node denotes a person, an edge represents that two person are involved in the same crime;

USairport [16], a network of flights between US airports in 2010. Each node is an airport, and each edge represents a connection from one airport to another;

Hamster [16]. This Network contains friendships and family links between users of the website hamster.com;

Figeys [16], a network of interactions between proteins in Humans (Homo sapiens), from the first large-scale study of protein–protein interactions in Human cells using a mass spectrometry-based approach;

CA-GrQc [18], a collaboration network from the e-print arXiv and covers scientific collaborations between authors papers submitted to General Relativity and Quantum Cosmology category;

HI-II-14, the corresponding **H**uman **I**nteractome dataset covering Space **II** and reported in 20**14**. Each node represents a distinct protein, each edge denotes the interaction between the corresponding proteins;

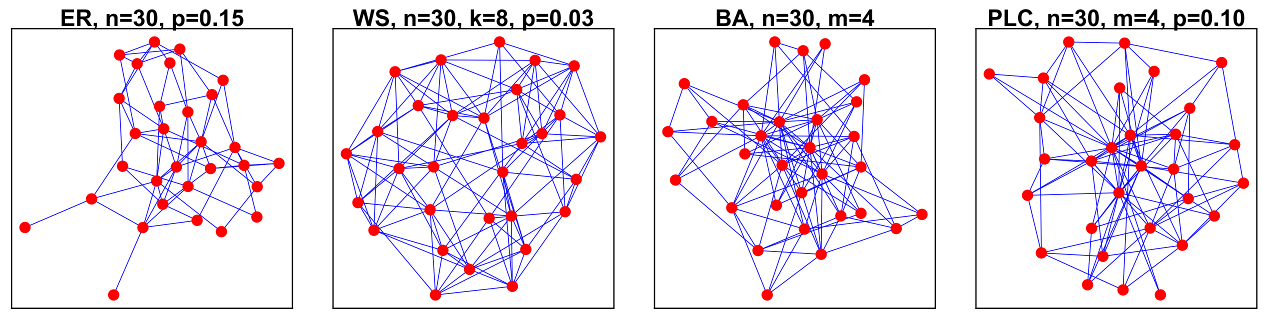

Fig. 3 Visualization of one instance for each type of random networks

Powergrid [16], a power grid network of the Western States of the United States of America. An edge represents a power supply line. A node is either a generator, a transformator or a substation;

CA-HepPh [18], a collaboration network from the e-print arXiv and covers scientific collaborations between authors papers submitted to High Energy Physics - Phenomenology category;

DBLP [16], a citation network of DBLP, a database of scientific publications such as papers and books. Each node in the network is a publication, and each edge represents a citation of a publication by another publication;

Cora [16], a citation network of Cora. Nodes represent scientific papers. An edge between two nodes indicates that the left node cites the right node;

Digg [16], a reply network of the social news website Digg. Each node in the network is a user of the website, and each edge denotes that a user replied to another user;

Email-Enron [20], the Enron email communication network which covers all the email communication within a data set of around half million emails. Each node is an email address, and an edge denotes at least one email communication;

Brightkite [16], a social network contains user–user friendship relations from Brightkite, a former location-based social network were user shared their locations. A node represents a user, and an edge indicates that a friendship exists between the user represented by the left node and the user represented by the right node;

Gnutella31 [19], a sequence of snapshots of the Gnutella peer-to-peer file sharing network from August 2002. Nodes represent hosts in the Gnutella network topology and edges represent connections between the Gnutella hosts;

Facebook [16], contains friendship data of a small subset of Facebook users. A node represents a user and an edge represents a friendship between two users;

Epinion [16], the trust network from the online social network Epinions. Nodes are users of Epinions and directed edges represent trust between the users;

Douban [16], a social network of Douban, a Chinese online recommendation site. A node represents a user of Douban and an edge represents a friendship between two users.

We treat all the networks as undirected ones and remove the self-loops. We extract the largest connected component. Basic statistics of the extracted networks are reported as Table 2.

We also draw the degree distributions for these networks in Fig. 4. We can see most real networks (except Corruption network) share an approximate scale-free structure, which presents a well-known resilience against random failures, but disintegrate rapidly under intentional attacks targeting key nodes [2].

Table 2 Basic statistics for real-world networks. Ordered by the number of nodes. Values are for the giant component of the network. **MSP** is the mean shortest path length, **CC** is the clustering coefficient, **Assor** is the assortativity, **PE** is the powerlaw exponent

Network	N	E	MAX DEG	AVG DEG	Diameter	MSP	CC	Assor	PE	Type
Corruption	309	3,281	86	21.24	7	2.99	0.9288	0.5324	5.13	Malicious
Crime	829	1,473	25	3.55	10	5.04	0.0058	−0.1645	2.65	Malicious
USairport	1,572	17,214	314	21.90	8	3.12	0.5048	-0.1134	3.07	Infrastructure
Hamster	2,000	16,098	273	16.10	10	3.59	0.5401	0.0227	2.67	Social
Figeys	2,217	6,418	314	5.79	10	3.84	0.0403	-0.3318	2.92	PPI
CA-GrQc	4,158	13,422	81	6.46	17	6.05	0.5569	0.6392	2.47	Collaboration
HI-II-14	4,165	13,087	286	6.28	11	4.16	0.0444	−0.2016	2.68	PPI
Powergrid	4,941	6,594	19	2.67	46	18.99	0.0801	0.0035	2.50	Infrastructure
CA-HepPh	11,204	117,619	491	21.00	13	4.67	0.6216	0.6295	2.67	Collaboration
DBLP	12,495	49,563	709	7.93	10	4.42	0.1178	-0.0461	3.22	Citation
Cora	23,166	89,157	377	7.70	20	5.74	0.2660	-0.0553	2.44	Citation
Digg	29,652	84,781	310	5.72	12	4.68	0.0054	0.0027	2.46	Communication
Email-Enron	33,696	180,811	1,383	10.73	11	4.03	0.5092	−0.1165	2.62	Communication
Brightkite	56,739	212,945	1,134	7.51	18	4.86	0.1734	0.0096	2.50	Social
Gnutella31	62,561	147,878	95	4.73	11	5.96	0.0055	−0.0927	2.91	Infrastructure
Facebook	63,392	816,831	1,098	25.77	15	4.31	0.2218	0.1768	2.92	Social
Epinions	75,877	405,739	3,044	10.69	15	4.40	0.1378	−0.0406	2.69	Social
Douban	154,908	327,162	287	4.22	9	5.10	0.0161	-0.1803	2.65	Social

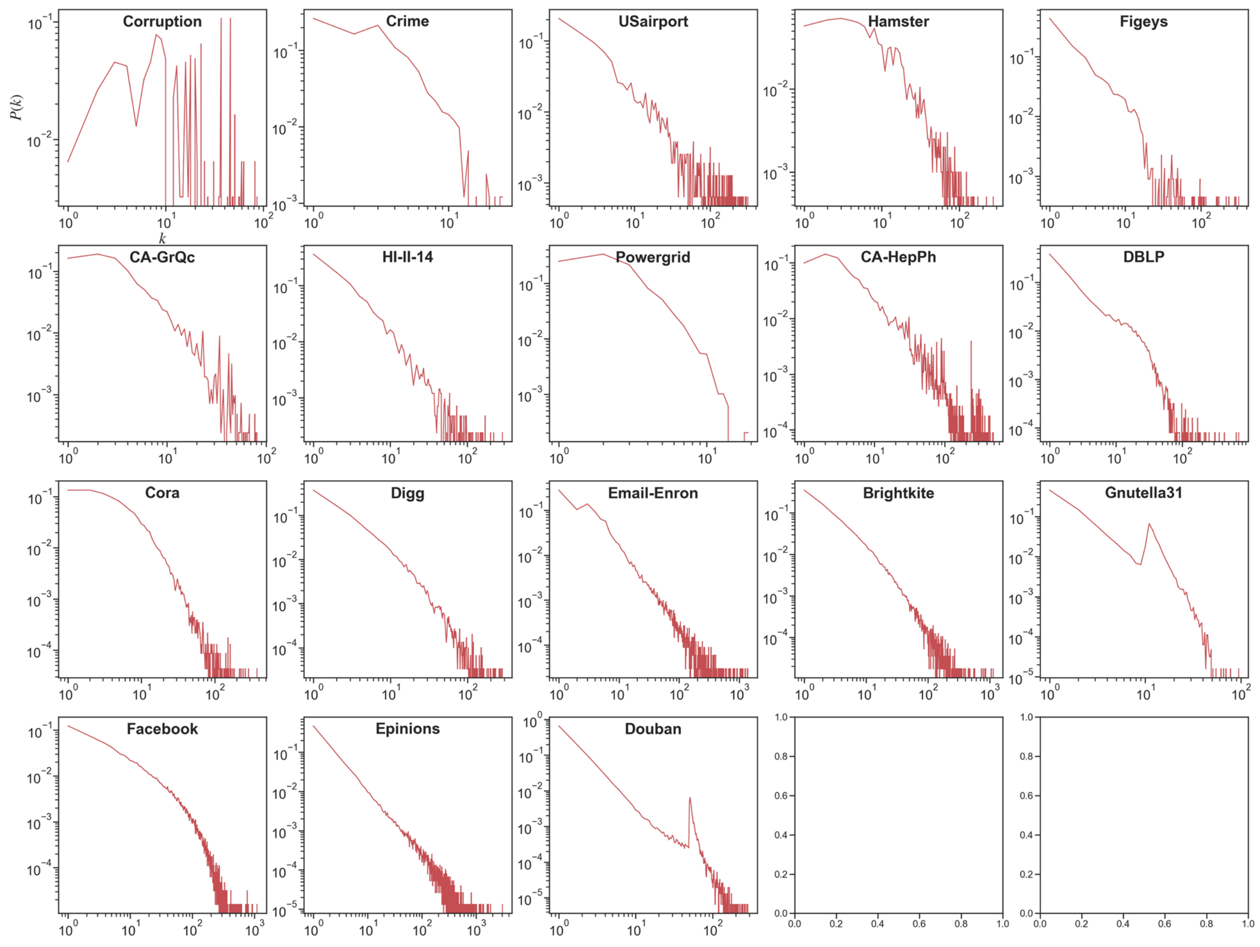

Fig. 4 Degree distribution of the real-world networks. X axis is the node degree, y axis is the number of nodes corresponding to that degree, and we choose double logarithmic coordinates to more intuitively determine whether it is scale-free

Results

In this section, we first demonstrate the effectiveness of the reinsertion technique on both synthetic graphs and real-world networks, then we explore the effects of different reinsertion techniques.

Synthetic results

We test all methods w/o the reinsertion technique on synthetic graphs randomly generated by four classic models introduced in "Synthetic graphs" section. For each model, we generate 100 graphs with the parameters in

Table 3 Comparison results (%) on synthetic graphs without reinsertion. Each result is averaged over 100 test instances. The result format is mean±variance. The bold ones indicate the best results for that network

R(No reinsert)	ER	WS	BA	PLC	Avg
Random	49.91±0.02	44.19±3.66	45.80±2.87	46.37±2.66	46.57±2.30
HDA	49.55±0.25	32.44±5.32	21.43±7.70	21.89±7.53	31.33±5.20
HBA	**49.47±0.29**	**25.55±7.59**	**19.96±7.65**	**20.23±7.52**	**28.80±5.76**
HCA	49.58±0.23	28.33±7.07	20.99±7.87	21.26±7.72	30.04±5.72
HPRA	49.55±0.25	33.10±4.88	22.46±7.77	22.92±7.64	32.01±5.13
CI	49.70±0.25	31.21±5.81	21.94±7.67	22.35±7.51	31.30±5.31
MinSum	49.45±0.31	32.43±5.09	20.58±7.70	21.40±7.36	30.96±5.12
BPD	49.58±0.24	33.69±4.60	22.75±7.63	23.21±7.49	32.31±4.99
CoreHD	49.56±0.26	32.48±5.33	21.01±7.76	21.72±7.50	31.19±5.21
GND	49.62±0.14	27.10±7.62	22.89±7.67	23.01±7.42	30.66±5.71

Table 4 Comparison results (%) on synthetic graphs with reinsertion. Each result is averaged over 100 test instances. The result format is mean±variance. The bold ones indicate the best results for that network

R(With reinsert)	ER	WS	BA	PLC	Avg
Random	48.97±0.44	33.12±4.24	27.24±7.17	28.14±7.56	34.37±4.85
HDA	**48.84±0.51**	31.36±4.59	**20.75±7.20**	21.21±7.12	30.54±4.86
HBA	48.92±0.46	33.18±4.28	25.41±7.47	25.90±7.79	33.35±5.00
HCA	48.87±0.48	31.72±4.43	20.90±7.24	21.27±7.13	30.69±4.82
HPRA	48.84±0.51	31.32±4.61	20.77±7.22	**21.16±7.13**	**30.52±4.87**
CI	49.67±0.29	**31.16±5.47**	21.68±7.53	22.12±7.41	31.16±5.17
MinSum	48.80±0.53	31.61±4.48	20.78±7.22	21.19±7.10	30.59±4.83
BPD	49.39±0.35	31.22±4.97	21.17±7.45	21.60±7.34	30.84±5.03
CoreHD	49.36±0.36	31.48±5.02	21.24±7.42	21.66±7.35	30.93±5.04
GND	49.20±0.33	31.84±5.67	21.67±7.62	22.10±7.60	31.20±5.30

Table 5 The promotion of *R* (%) on synthetic graphs with reinsertion. Each result is averaged over 100 test instances. The result format is mean±variance. The bold ones indicate the best results for that network

Promotion	ER	WS	BA	PLC	Avg
Random	**1.88±0.86**	**24.96±8.41**	**41.23±12.72**	**40.01±13.75**	**27.02±8.94**
HDA	1.43±0.53	2.77±7.41	2.54±2.57	2.67±2.27	2.35±3.19
HBA	1.13±0.36	-43.55±54.37	-33.61±19.64	-33.95±21.25	-27.50±23.91
HCA	1.44±0.53	-18.16±30.91	-1.07±4.68	-1.73±5.42	-4.88±10.39
HPRA	1.45±0.55	5.12±7.94	7.50±2.09	7.76±2.09	5.46±3.17
CI	0.05±0.13	-0.11±2.34	1.08±0.92	0.96±0.84	0.49±1.06
MinSum	1.33±0.46	2.16±4.85	-2.38±4.29	0.52±3.16	0.41±3.19
BPD	0.40±0.26	7.62±3.18	7.59±2.70	7.71±3.35	5.83±2.37
CoreHD	0.41±0.28	2.95±3.00	-2.18±3.55	-0.07±2.38	0.28±2.30
GND	0.85±0.52	-22.46±21.35	5.97±4.67	4.96±4.20	-2.67±7.68

Table 6 Time (/s) comparison of different methods on synthetic graphs. Each result is averaged over 100 test instances. The result format is mean±variance. The bold ones indicate the best results for that network

Time/s	ER	WS	BA	PLC	Avg
HDA	**0.01±0.00**	**0.00±0.00**	**0.00±0.00**	**0.00±0.00**	**0.00±0.00**
HBA	426.71±271.03	41.03±20.11	31.53±18.58	33.42±20.92	133.17±82.66
HCA	311.23±201.09	20.21±9.83	14.10±8.49	14.83±9.47	90.09±57.22
HPRA	0.77±0.24	0.53±0.13	0.45±0.12	0.46±0.13	0.55±0.15
CI	8.67±6.30	0.02±0.01	0.03±0.03	0.03±0.02	2.19±1.59
MinSum	22.21±9.92	2.28±0.47	1.93±0.81	2.69±0.80	7.28±3.00
BPD	148.21±100.97	0.85±0.46	0.94±0.66	1.02±0.68	37.76±25.69
CoreHD	3.06±1.14	0.12±0.04	0.18±0.08	0.20±0.09	0.89±0.34
GND	0.86±0.35	0.18±0.10	0.15±0.08	0.14±0.09	0.33±0.15

Table 7 Comparison results (%) on real-world networks without reinsertion. The bold result is the best one of that network

R(No reinsert)	Random	HDA	HPRA	CI	MinSum	BPD	CoreHD	GND
Corruption	38.50	8.48	8.81	16.42	33.27	33.74	8.37	**6.37**
Crime	38.64	**11.51**	11.83	36.77	17.95	36.59	36.74	37.61
USairport	43.73	11.39	10.15	13.02	13.12	13.01	12.44	**8.94**
Hamster	44.40	19.00	16.02	16.36	23.27	22.77	20.23	**15.58**
Figeys	37.93	**3.13**	3.16	3.38	4.27	4.46	3.65	3.90
CA-GrQc	37.41	10.81	8.56	9.37	12.30	12.38	11.33	**7.48**
HI-II-14	40.60	**5.75**	5.89	22.41	7.28	23.14	23.18	23.80
Powergrid	21.47	5.25	5.90	5.23	5.58	4.74	5.49	**2.20**
CA-HepPh	42.41	18.55	14.78	16.71	20.84	20.71	19.31	**13.14**
DBLP	41.32	10.65	**10.00**	11.16	12.29	12.21	11.98	11.07
Cora	42.87	14.85	14.92	14.41	16.82	16.65	15.36	**10.98**
Digg	40.16	**8.75**	9.12	29.00	26.39	26.35	26.53	27.72
Email-Enron	39.63	4.53	**4.18**	18.97	6.30	19.20	19.88	16.78
Brightkite	39.30	8.73	8.79	**8.47**	9.55	22.07	9.03	9.05
Gnutella31	39.66	11.47	**10.88**	27.24	11.74	26.44	26.62	26.95
Facebook	45.81	27.24	**27.06**	41.68	27.91	41.22	41.58	41.91
Epinions	38.61	**5.19**	5.20	18.26	6.11	18.26	18.47	18.39
Douban	36.73	2.38	**2.20**	2.76	2.89	2.87	2.93	2.63
Avg	39.40	10.43	**9.86**	17.31	14.33	19.82	17.40	15.81

Table 8 Comparison results (%) on real-world networks with reinsertion. The bold result is the best one of that network

R(With reinsert)	Random	HDA	HPRA	CI	MinSum	BPD	CoreHD	GND
Corruption	12.78	**11.66**	11.73	18.40	18.63	18.34	18.82	21.14
Crime	14.19	**11.33**	11.41	36.47	14.18	36.32	36.12	36.60
USairport	14.49	9.85	9.76	11.68	9.61	**9.58**	9.65	9.82
Hamster	18.31	**14.71**	15.16	15.57	16.23	16.38	16.66	16.24
Figeys	3.99	3.14	**3.13**	3.28	3.44	3.39	3.43	3.76
CA-GrQc	10.31	**8.59**	8.66	8.70	8.85	8.67	8.75	7.70
HI-II-14	7.00	**5.68**	5.70	22.00	5.76	22.75	22.76	23.07
Powergrid	7.93	6.94	7.16	4.90	5.02	4.92	5.08	**4.44**
CA-HepPh	15.40	14.41	14.49	14.53	15.39	15.14	15.19	**13.98**
DBLP	10.42	**8.75**	8.75	8.94	10.18	10.05	10.15	9.79
Cora	17.00	13.48	13.49	**13.11**	14.05	13.79	14.04	13.83
Digg	10.96	**8.59**	8.60	28.21	26.28	26.24	26.24	26.94
Email-Enron	6.01	3.96	**3.95**	17.91	4.25	18.46	18.68	16.20
Brightkite	10.18	8.20	8.21	**7.69**	8.41	22.06	8.35	8.41
Gnutella31	11.20	**10.07**	10.07	25.98	10.83	25.87	25.86	25.46
Facebook	25.38	22.39	**22.37**	40.67	25.65	40.67	40.68	39.96
Epinions	5.70	4.96	**4.94**	17.95	4.97	18.18	18.18	18.00
Douban	2.72	**2.09**	2.09	2.17	2.38	2.36	2.36	2.31
Avg	11.33	**9.38**	9.43	16.56	11.34	17.40	16.72	16.54

Table 9 The promotion of *R* (%) on real-world networks with reinsertion. The bold result is the best one of that network

Promotion of *R*	Random	HDA	HPRA	CI	MinSum	BPD	CoreHD	GND
Corruption	**66.81**	-37.50	-33.14	-12.06	44.00	45.64	-124.85	-231.87
Crime	**63.28**	1.56	3.55	0.82	21.00	0.74	1.69	2.69
USairport	**66.86**	13.52	3.84	10.29	26.75	26.36	22.43	-9.84
Hamster	**58.76**	22.58	5.37	4.83	30.25	28.06	17.65	-4.24
Figeys	**89.48**	-0.32	0.95	2.96	19.44	23.99	6.03	3.59
CA-GrQc	**72.44**	20.54	-1.17	7.15	28.05	29.97	22.77	-2.94
HI-II-14	**82.76**	1.22	3.23	1.83	20.88	1.69	1.81	3.07
Powergrid	**63.06**	-32.19	-21.36	6.31	10.04	-3.80	7.47	-101.82
CA-HepPh	**63.69**	22.32	1.96	13.05	26.15	26.90	21.34	-6.39
DBLP	**74.78**	17.84	12.50	19.89	17.17	17.69	15.28	11.56
Cora	**60.35**	9.23	9.58	9.02	16.47	17.18	8.59	-25.96
Digg	**72.71**	1.83	5.70	2.72	0.42	0.42	1.09	2.81
Email-Enron	**84.83**	12.58	5.50	5.59	32.54	3.85	6.04	3.46
Brightkite	**74.10**	6.07	6.60	9.21	11.94	0.05	7.53	7.07
Gnutella31	**71.76**	12.21	7.44	4.63	7.75	2.16	2.85	5.53
Facebook	**44.60**	17.80	17.33	2.42	8.10	1.33	2.16	4.65
Epinions	**85.24**	4.43	5.00	1.70	18.66	0.44	1.57	2.12
Douban	**92.59**	12.18	5.00	21.38	17.65	17.77	19.45	12.17
Avg	**71.56**	5.88	2.11	6.21	19.85	13.36	2.27	-18.02

Table 1, and report the values of mean and standard variance results. Table 3 shows the comparison results of Eq. 1 without reinsertion, we can clearly see that HBA the best across different types of networks, which is widely validated by previous research [14, 27], since HBA adaptively removes the highest betweenness nodes, which are key to the whole network connectivity. HCA, which adaptively removes the highest closeness nodes, also performs excellently due to the similar reasons. However, considering the high computational costs of these two methods (Table 6), they are not practical in large or even medium scale networks. We can also see in methods achieve good results in ER graphs, since these graphs are purely random ones that there are no 'critical' nodes that determine the graph connectivity.

In Table 4, we enhance each method with the reinsertion technique introduced in "Reinsertion technique" section, and report the refined results, and we also show the

Table 10 Time (/s) comparison of different methods on real-world networks. The bold result is the best one of that network

Time/s	HDA	HPRA	CI	MinSum	BPD	CoreHD	GND
Corruption	**0.00**	0.11	0.02	3.13	2.00	0.08	0.18
Crime	**0.00**	0.55	0.01	1.47	0.30	0.09	0.11
USairport	**0.01**	1.96	0.61	13.20	17.00	1.20	0.55
Hamster	**0.01**	4.74	0.44	13.89	7.00	0.90	0.68
Figeys	**0.00**	1.57	0.06	5.06	2.00	0.41	0.13
CA-GrQc	**0.04**	18.33	0.07	12.63	3.00	0.46	1.29
HI-II-14	**0.02**	10.66	0.49	10.51	9.00	1.98	1.32
Powergrid	**0.05**	23.06	0.01	5.90	1.00	0.21	1.48
CA-HepPh	**0.31**	170.96	13.90	138.90	140.00	8.50	20.69
DBLP	**0.13**	97.31	2.81	46.97	21.00	3.33	6.30
Cora	**1.07**	593.75	2.12	97.97	22.00	4.91	189.27
Digg	**1.12**	702.55	10.99	103.33	31.00	11.52	55.01
Email-Enron	**1.63**	876.55	70.74	242.27	86.00	36.20	35.70
Brightkite	**4.47**	2974.39	32.63	298.20	4.00	20.13	181.76
Gnutella31	**3.05**	2509.09	9.38	172.59	41.00	11.87	386.31
Facebook	**9.03**	6706.60	2723.23	1044.96	839.00	214.83	4614.54
Epinions	**5.76**	4115.42	668.43	522.51	254.00	142.92	233.14
Douban	**2.88**	3832.00	87.38	337.04	79.00	30.20	186.03
Avg	**1.64**	1257.76	201.30	170.58	86.57	27.21	328.58

Table 11 Average promotion of *R* on synthetic graphs for different reinsertion techniques. Each result is averaged over all test graphs (including four types of graphs, and 100 graphs for each type), and the result format is mean±variance. The bold result is the best one for that method

Avg Promotion of *R*	Reinsert_I	Reinsert_II	Reinsert_III
Random	27.02±8.94	32.73±10.87	**33.36±10.74**
HDA	2.35±3.19	9.96±5.16	**10.67±4.81**
HBA	-27.50±23.91	-11.47±7.94	**-10.53±7.82**
HCA	-4.88±10.39	5.08±1.80	**5.82±1.81**
HPRA	5.46±3.17	12.68±5.68	**13.48±5.43**
CI	**0.49±1.06**	-0.52±1.07	-0.14±0.93
MinSum	0.41±3.19	8.17±5.41	**8.86±5.11**
BPD	5.83±2.37	4.73±2.95	**5.86±3.34**
CoreHD	0.28±2.30	-0.96±2.81	**0.35±2.77**
GND	**-2.67±7.68**	-4.50±7.57	-3.04±7.11

promotion (Eq. 2) after adding the reinsertion in Table 5. We can see that most methods (except for HBA, HCA and GND) get improved after using reinsertion, and on average, HPRA (reinserted) performs the best among all. We also observe two interesting things: i) The best performed HBA gets deteriorated greatly when utilized with reinsertion, however, even the best result for reinserted methods (HPRA) cannot beat the vanilla HBA (Table 3). This indicates that the vanilla HBA has achieved the close-to-optimal performance for the network dismantling problem, at which the reinsertion is no longer a refinement, but a hindrance; ii) The pure Random strategy gets greatly improved with reinsertion, making the reinserted random strategy be close to those manually-designed state-of-the-arts.

$$promotion = \frac{(R_{original} - R_{reinsert})}{R_{original}} \tag{2}$$

However, if taking account of running time, we find actually the simple heuristic HDA achieves the best balance between effectiveness and efficiency (Tables 3, 4 and 6). The reinserted HDA is only 1.74% worse than the best result (vanilla HBA), while is hundreds of times faster (Table 6). Note that we do not list the time for Random strategy, since it basically takes no time to obtain a random solution.

Real-world results

Now we will see the effects of reinsertion on real-world networks. Since HBA and HCA are computationally prohibitive on medium or large networks (e.g., HBA takes over 5 days to finish computation on the Cora network, with 23,166 nodes and 89,157 edges.), we do not compare with them in this section.

Table 7 shows the results of vanilla methods without reinsertion. We can see that HDA, HPRA and GND performs relatively better than other methods, and HPRA is the best (0.0986) among all, and followed by HDA (0.1043). Table 8 gives the results after reinsertion, and Table 9 shows the promotion results. Consistent with the observations from synthetic results, most methods get improvements for different levels, with the refinement of reinsertion. For example, the random strategy obtains an average 71.56% gain (Table 9) with reinsertion, making it even beat the state-of-the-art MinSum strategy (Table 8). Among the reinserted methods, HDA achieves the highest performance with an average 0.0938 (Table 8) robustness score (Eq. 1). However, GND is deteriorated on some networks when refined with reinsertion (Table 8), the reason behind remains to be explored. When considering the execution, HDA is far more efficient than the other ones, e.g., it is about 767 times faster than HPRA, which is very close to HDA in effectiveness.

Effects of different reinsertion strategies

We have observed the impressive gains brought by the reinsertion technique in "Synthetic results" and "Real-world results" sections, now we may ask: *Is the reinsertion in "Reinsertion technique" section the best one? Does there exist more effective reinsertion methods?* In this section, we try to answer this question by exploring other potential reinsertion techniques (Table 10).

We name the previous reinsert method as **Reinsert_I**, and here we propose two other ones, and call them **Reinsert_II** and **Reinsert_III** respectively. Basically, the general reinsertion technique is to add back one of the removed node (together with the adjacent edges), chosen based on some *criteria*, until all nodes are back in the network. Different reinsertion methods define different *criteria*, based on which, we define the following three reinsertion strategies:

- **Reinsert_I:** The *criteria* is once reinserted, it joins the smallest number of clusters;
- **Reinsert_II:** The *criteria* is once reinserted, it joins the clusters of smallest sizes;
- **Reinsert_III:** The *criteria* is once reinserted, it joins the clusters minimizing the multiply of both numbers and sizes;

In Fig. 2, each node is assigned an index $c(i)$ given by the criteria specified by the reinsertion technique. For **Reinsert_I**, $c(red) = 2, c(blue) = 4, c(green) = 3$, then the red node is reinserted; for **Reinsert_II**, $c(red) = 10, c(blue) = 5, c(green) = 6$, then the blue node is reinserted; for **Reinsert_III**, $c(red) = 20, c(blue) = 20, c(green) = 18$, then the green node is reinserted. After that, the $c(i)$s are recalculated and the new node with smallest $c(i)$ is found and reinserted. Repeat these steps until the end. As a consequence, different reinsertion strategies determines different nodes to be reinserted first, leading to different refinement results. To decide which one is better in

Table 12 Average promotion of *R* on real-world networks for different reinsertion techniques. Each result is averaged over all test networks (total 18 real-world networks), and the bold ones are the best results for that method

Avg Promotion of *R*	Reinsert_I	Reinsert_II	Reinsert_III
Random	71.56	77.31	**78.24**
HDA	5.88	21.71	**23.98**
HPRA	2.11	19.14	**21.45**
CI	**6.21**	4.46	6.01
MinSum	19.85	16.72	**20.59**
BPD	13.36	12.33	**14.49**
CoreHD	2.27	6.92	**9.79**
GND	-18.02	-9.43	**-8.16**

practice, we compare the average performance promotion for each method on both synthetic graphs and real-world networks (Tables 11 and 12).

It can be clearly observed in Tables 11 and 12 that **Reinsert_III** achieves the most promotions for most methods (except CI) on both synthetic and real-world networks, compared to other two reinsertion strategies, and excels to a significant extent to the current strategy **Reinsert_I**. For CI method, **Reinsert_I** tends to be more effective. All the three reinsertion strategies fail in HBA and GND.

To illustrate the effects of these three strategies more intuitively, we draw the robustness curve of CA-GrQc network for different methods with different reinsertions in Fig. 5, which is plotted with horizontal axis being the fraction of removed nodes, and vertical axis being the remaining giant connected component size. Actually, the value of Eq. 1 approximates the area under the robustness curve. The figure clearly shows that the reinsertion greatly helps reduce the area under the curve, compared to the original method, and **Reinsert_III** is among the most effective one, while all the reinsertions produce negative effects on the GND method.

Conclusion

In this paper, we, for the first time, systematically explore the effects of reinsertion techniques for the network dismantling problem. Previous research tend to use their reinserted results to compare with other un-reinserted baseline methods, which may mislead us in the selection of the real best dismantling strategy for applications at hand. We conduct comprehensive ablation studies on both synthetic graphs generated by four classical random network models, i.e., ER, WS, BA and PLC, and 18 real-world networks across seven different domains and with different scales, and the results show that: i) HBA (no reinsertion) is the most effective network dismantling strategy, however, it can only be applicable in small scale networks; ii) HDA (with reinsertion) achieves the best balance between effectiveness and efficiency. It is surprising that such a simple heuristic method would beat most state-of-the-art methods if enhanced with reinsertion techniques; iii) The reinsertion technique helps improve the performance for most current methods, except for HBA, HCA and GND (on small-world type graphs); iv) Reinsert_III, which determines the node based on that it joins the clusters minimizing the multiply of both numbers and sizes, is the most effective reinsertion strategy for most methods (except for CI, where Reinsert_I suits best). We believe the results in this paper could provide as a reference for choosing and designing the most effective strategy for realistic network dismantling applications.

However, we still lack a deep understanding about why such a simple reinsertion technique works so well for the network dismantling problem, which would be a very meaningful future research topic to be explored. We will later release the codes and data to support the research in this direction.

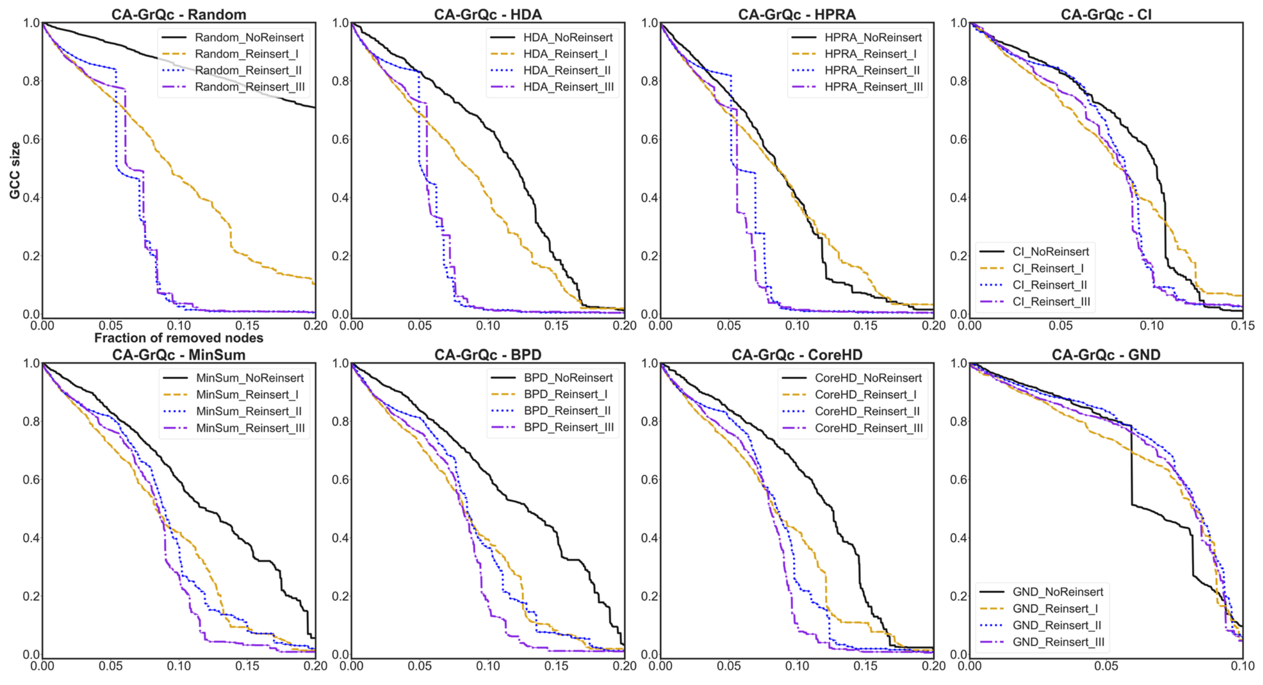

Fig. 5 Robustness curve of CA-GrQc for different reinsertion techniques

Acknowledgments
We thank Yuan Liu for her valuable discussions.

Authors' contributions
CF and YF initiated the project. CF and YF, XB designed and managed the project. CF and LZ performed calculations. All authors analyzed the results, wrote the manuscript, and edited the manuscript. All author(s) read and approved the final manuscript.

Authors' information
CF, LZ, YF, JH and ZL are all affiliated with College of Systems Engineering, National University of Defense Technology. CF and LZ are both Ph.D. candidates. YF is an associate professor, JH and ZL are both professors. XB is a professor in School of Systems Science and Engineering, Sun Yat-sen University.

Funding
This work was supported by CSC scholarship offered by Chinese Scholarship Council and NSFC-71701205.

Availability of data and materials
Upon reasonable requests, all code/data used in the analysis will be available to any researcher for purposes of reproducing or extending the analysis.

Competing interests
The authors declare that they have no competing interests.

Author details
[1]College of Systems Engineering, National University of Defense Technology, Changsha, Hunan, China. [2]School of Systems Science and Engineering, Sun Yat-sen University, Guangzhou, Guangdong Province, China.

Received: 30 June 2019 Accepted: 3 April 2020
Published online: 07 May 2020

References
1. Albert R, Barabási AL (2002) Statistical mechanics of complex networks. Rev Mod Phys 74(1):47
2. Albert R, Jeong H, Barabási AL (2000) Error and attack tolerance of complex networks. Nature 406(6794):378
3. Barabási AL, Albert R (1999) Emergence of scaling in random networks. Science 286(5439):509–512
4. Bavelas A (1950) Communication patterns in task-oriented groups. J Acoust Soc Am 22(6):725–730
5. Braunstein A, Dall'Asta L, Semerjian G, Zdeborová L (2016) Network dismantling. Proc Natl Acad Sci 113(44):12,368–12,373
6. Brin S, Page L (1998) The anatomy of a large-scale hypertextual web search engine. Comput Netw ISDN Syst 30(1-7):107–117
7. Cohen R, Erez K, Ben-Avraham D, Havlin S (2001) Breakdown of the internet under intentional attack. Phys Rev Lett 86(16):3682
8. ERDdS P R&wi (1959) On random graphs i. Publ Math Debrecen 6:290–297
9. Fan C, Xiao K, Xiu B, Lv G (2014) A fuzzy clustering algorithm to detect criminals without prior information. In: Proceedings of the 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. IEEE Press. pp 238–243
10. Fan C, Liu Z, Lu X, Xiu B, Chen Q (2017) An efficient link prediction index for complex military organization. Phys A Stat Mech Appl 469:572–587
11. Fan C, Zeng L, Ding Y, Chen M, Sun Y, Liu Z (2019) Learning to identify high betweenness centrality nodes from scratch: A novel graph neural network approach. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management. pp 559–568
12. Freeman LC (1977) A set of measures of centrality based on betweenness. Sociometry: 35–41
13. Holme P, Kim BJ (2002) Growing scale-free networks with tunable clustering. Phys Rev E 65(2):026,107
14. Holme P, Kim BJ, Yoon CN, Han SK (2002) Attack vulnerability of complex networks. Phys Rev E 65(5):056,109
15. Kempe D, Kleinberg J, Tardos É (2003) Maximizing the spread of influence through a social network. In: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM. pp 137–146
16. Kunegis J (2013) Konect: the koblenz network collection. In: Proceedings of the 22nd International Conference on World Wide Web. ACM. pp 1343–1350
17. Lalou M, Tahraoui MA, Kheddouci H (2018) The critical node detection problem in networks: a survey. Comput Sci Rev 28:92–117
18. Leskovec J, Krevl A (2014) SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data
19. Leskovec J, Kleinberg J, Faloutsos C (2007) Graph evolution: Densification and shrinking diameters. ACM Trans Knowl Discov Data (TKDD) 1(1):2
20. Leskovec J, Lang KJ, Dasgupta A, Mahoney MW (2009) Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. Internet Math 6(1):29–123
21. Morone F, Makse HA (2015) Influence maximization in complex networks through optimal percolation. Nature 524(7563):65
22. Mugisha S, Zhou HJ (2016) Identifying optimal targets of network attack by belief propagation. Phys Rev E 94(1):012,305
23. Pastor-Satorras R, Vespignani A (2001) Epidemic spreading in scale-free networks. Phys Rev Lett 86(14):3200
24. Ren XL, Gleinig N, Helbing D, Antulov-Fantulin N (2018) Generalized network dismantling. arXiv preprint arXiv:180101357
25. Ren XL, Gleinig N, Helbing D, Antulov-Fantulin N (2019) Generalized network dismantling. Proc Natl Acade Sci 116(14):6554–6559
26. Ribeiro HV, Alves LG, Martins AF, Lenzi EK, Perc M (2018) The dynamical structure of political corruption networks. J Complex Netw 6(6):989–1003
27. Schneider CM, Mihaljev T, Herrmann HJ (2012) Inverse targeting—an effective immunization strategy. EPL (Europhys Lett) 98(4):46,002
28. Von Luxburg U (2007) A tutorial on spectral clustering. Stat Comput 17(4):395–416
29. Wandelt S, Sun X, Feng D, Zanin M, Havlin S (2018) A comparative analysis of approaches to network-dismantling. Sci Rep 8(1):13,513
30. Watts DJ, Strogatz SH (1998) Collective dynamics of 'small-world' networks. Nature 393(6684):440
31. Zdeborová L, Zhang P, Zhou HJ (2016) Fast and simple decycling and dismantling of networks. Sci Rep 6:37,954