# Ensemble approach for generalized network dismantling

Xiao-Long Ren and Nino Antulov-Fantulin

ETH Zurich, Zrich CH-8092, Switzerland To whom correspondence should be addressed: anino@ethz.ch

**Abstract.** Finding a set of nodes in a network, whose removal fragments the network below some target size at minimal cost is called network dismantling problem and it belongs to the NP-hard computational class. In this paper, we explore the (generalized) network dismantling problem by exploring the spectral approximation with the variant of the power-iteration method. In particular, we explore the network dismantling solution landscape by creating the ensemble of possible solutions from different initial conditions and a different number of iterations of the spectral approximation.

Keywords: network dismantling, spectral partitioning, robustness

### 1 Introduction

The process of network (graph) fragmentation by removing nodes or edges has a long history [1,2,3,4,5,6] due to its practical relevance for maintaining the robustness of real-world systems [7,8,9,10,11], containing contagion processes [12,13,14]or identification of node importance [15]. In case of vertex or edge separators, we want to find a small separator S whose removal results in the partition to two roughly equal size [16] sets. Finding the minimum vertex or edge separator for general graphs is an NP-hard problem [4], and it was approximated by different methods of linear programming [17], semidefinite programming [18,19,20], and spectral partitioning [1,3,5].

In this paper, we will study the network dismantling problem [21,22,23,24,25]. A set S is called a C-dismantling set if the largest/giant connected component (GCC) of a network contains at most C nodes after removing the nodes in set S [21,22]. Finding a minimum C-dismantling set is called network dismantling problem [26]. Similarly, for a given network G(V, E) with nodal costs  $W = (w_1, \ldots, w_{|V|})$ , the generalized network dismantling [27] aims to find a set of nodes  $S(G, W, C) \subseteq V$  with the minimum dismantling cost, which will result in a fragmentation of the network into components of size at most C. The network dismantling problem belongs to the NP-hard class [26] and can not have a fully polynomial-time approximation scheme (FPTAS) [27] because of the theorem about the hardness approximating minimum vertex cover [28]. This has motivated us to explore the ensemble of the dismantling solutions, instead of focusing on the (over)optimization within the vicinity of one potential point in the optimization function. Furthermore, our approach is also motivated by the studies of analyzing energy landscapes of highly non-linear optimization (loss) functions in machine learning [29] and modularity maximization in network science [30,31]. In this paper, the exploration of dismantling landscape will be done by different initial conditions in the spectral partitioning method of generalized network dismantling method [27].

# 2 Network dismantling approaches

To solve the network dismantling problems, many efforts have been devoted recently. We will briefly introduce several representative algorithms we compared in this paper below.

Mugisha and Zhou [32] related the network dismantling problem to the feedback vertex set problem and applied the belief propagation-guided decimation (**BPD**) algorithm [33] to solve it. BPD algorithm is a loop-focused global algorithm which removes the nodes with the highest probability to break most loops in the network.

Braunsteina *et al.* [26] introduced a very efficient algorithm, **Min-Sum**, which consisted of three stages: (1) Using Min-Sum message passing to break all the loops in the network, then only trees are left. (2) Breaking all the trees whose size are bigger than the target dismantling size (threshold). (3) Greedily reinsert [34] the removed nodes that had been removed from the network in the previous two stages.

More recently, Ren *et al.* [27] studied the generalized network dismantling problem, which aims at finding a set of nodes with minimal dismantling cost. The dismantling cost can be any arbitrary non-negative real values. To solve this expanded problem, they proposed the **GND** algorithm which is based on the iterative node-weighted spectral approximation and a fine-tuning weighted vertex cover method [35]. Please find more details in Fig. 1 and Section 3.

Fan *et al.* [36] reformulated the network dismantling problem as a Markov decision process and employed deep reinforcement learning to train the **GraphDQN** agent to efficiently solve the problem. To the best of our knowledge, this is the first practice to solve the (generalized) network dismantling problem by using deep reinforcement learning approach.

In addition to the algorithms introduced above, there are also many other commonly used algorithms, such as equal graph partitioning (EGP) [37], Collective Influence (CI) [34], CoreHD [38], and so on [39]. Detailed comparisons of these algorithms can be found in ref. [27,36].

# 3 Ensemble-GND algorithm

More iterations or more random tries? The detailed procedure of the standard GND algorithm was elaborated in the paper [27]. The standard GND



**Fig. 1.** A brief explanation of the procedure of the standard GND algorithm, see more details in ref. [27]. When the size of the gaint connected components (GCC) is smaller than the target size, the algorithm will stop to remove more nodes and start to reinsert the removed nodes. This figure comes from ref. [27].

algorithm uses a variant of the power-iteration method to calculate the eigenvector of the second smallest eigenvalue of the weighted Laplacian matrix of the network. The spectral approximation uses the deterministic initialization with the pseudorandom Mersenne Twister generator [40] with default seed. The authors showed that in every bisection, it usually takes  $P = O(log(n) * \sqrt{(log(n))})$  iterations to get an effective eigenvector so that it can get a good partition. In particular, they have used  $P = 30 * log(n) * \sqrt{(log(n))}$  iterations.

The following question arises: Is it possible to obtain better results by allowing more iterations in the power method? Here we test the results when the number of iterations is D \* P (other conditions and parameters keep the same). The result on Petster-hamster network dataset [41] is shown below in Fig. 2. The blue curves in this figure are the standard/original GND(R) algorithms published in paper [27]. The green curves are the results of the tested procedure with D = 1000 times more iteration. This result suggests that more iterations in the power method doesn't necessary produce better dismantling result in the GND algorithm.

It is not surprising that being more accurate in approximating eigenvectors does not lead to better dismantling solution. After all, the problem is NP-hard, and the spectral formulation is just the integer relaxation [5] without strict bounds on the optimality. To contrast, we also tested another approach. As we know, every time before calculating the eigenvector in every bisection, the GND algorithm need to produce an initial vector v. Then after  $P = 30 * log(n) * \sqrt{(log(n))}$  iterations the power method will get the approximation of the eigenvector. According to this approximation, all the nodes in the GCC of the network will be partition into two parts, M and  $\overline{M}$ . Then the weighted vertex cover method will be applied and GND can get the set of nodes

4 Xiao-Long Ren, Nino Antulov-Fantulin



Fig. 2. Comparison of the standard GND(R) algorithms with its variants. The blue curves are the standard GND and GNDR algorithms published in ref. [27]. The green curves are the standard GND(R) algorithms with D = 1000 times more iterations in the power method when computing the eigenvector in every bisection.

that should be removed in this bisection. As we can see, the result of the bisection is based on the the initial vector v which is always deterministic, due to the default seed of the pseudorandom number generator. Default seed is used for the reproducibility purposes.

Alternative approach of using single initialization with D \* P iterations is to use K different initialization with P iterations, which we call ensemble approach. In order to have the same run-time complexity, we will fix K = D. In the ensemble approach, we will produce K different dismantling solutions  $S_1, S_2, \dots, S_K$ , and take the one with the with the **minimum cost** 

$$S^* = \min\{S1, S2, ..., S_K\}.$$

The statistical behavior of the minimum cost is given by the extreme value distribution, however, in this paper we only use deterministic approach. Different initializations are produced with the pseudorandom number generator with default seed, which results with deterministic method. This approach has the same computational complexity and similar running time with the method we tried above (green curves). The results of this approach are the red curves shown in Fig. 2. We can see that this variant GND and its GNDR algorithm (red curves) have much better performances than the standard GND and GNDR algorithms. In Fig. 3, we show the variability of the ensemble approach that was explored with K = 10 different initializations.



Fig. 3. Variability of the ensemble approach of GND. (A)-(C) show numerical results for 10 different initializations: (A) with P iterations for each initialization, (B) with 200 \* P iterations for each initialization, (C) with 500 \* P iterations for each initialization, where  $P = 30 * \log(n) * \sqrt{\log(n)}$ . When number of iterations D \* P exceeds D = 500, we no longer see the variability from different initializations. (D) We measure the difference in the dismantling performance  $GCC_{500*P}(c) - GCC_P(c)$ , where  $GCC_x(c)$  denotes the GND algorithm with x spectral approximation iterations for cost c. The graphic shows the histogram of differences in GCC over all possible costs for different seeds. We observe that the majority of differences is positive, which implies having a smaller GCC for the same cost for setting with P iterations.

Fine-tuning of the initial partition in every bisection of the GND algorithm. In the standard GND algorithm, after getting the eigenvector corresponding to the second smallest eigenvalue of the weighted Laplacian matrix, all the nodes will be partition into two groups, M and  $\overline{M}$ , according to their values in the eigenvector. More specifically, the nodes with a value smaller than 0 will be put in group M, else group  $\overline{M}$ . After this, the links between the two groups should be removed to partition the network into two disconnected parts

#### 6 Xiao-Long Ren, Nino Antulov-Fantulin

(see Fig. 1A(4) and ref. [27]). Then the weighted vertex cover method will be applied to fine-tuning the dismantling set.



**Fig. 4.** An example of partitioning the nodes in Petster-hamster network [41] into two groups according to their values in the eigenvector. The green nodes have values smaller than 0 (group M) while the red nodes have values equal to or bigger than 0 (group  $\overline{M}$ ).

However, the initial partition of the two groups exactly according to nodes? value in its eigenvector is not always a perfect choice. We exemplify this fact in Fig. 4, which shows an instance of partitioning the nodes in Petster-hamster network into two groups according to their values in its eigenvector. All the green nodes have values smaller than 0 (group M) while all the red nodes have values equal to or bigger than 0 (group  $\overline{M}$ ). Thus the edges between the two groups should be removed to make the group disconnected. Then the vertex cover method will be employed to find the fine-tuning dismantling set of nodes based on this step. Please note that there are N = 2,000 nodes in the Petsterhamster network and the target size of the dismantling is C = 1% \* N = 20. In this example, removing nodes  $\{919, 1274, 1049, 1048, 1051\}$  does not reduce the size of GCC below the target but is increasing the dismantling cost. To improve the performance of the standard GND algorithm, the partition of the nodes should be adjusted in the above examples. Thus we will adjust the ascription (i.e., M or M) of a node if all of its neighbors are belonging to another group. For example, the nodes {919, 1049, 1048, 1051} will be adjusted to red while the node {1274} will become green after adjustment and all these nodes do not need to be removed. The **spectral fine-tuning** can be formalized as the following

rule. For an arbitrary node v belonging to group M with cardinality |M| > 1, if all its neighbors belong to the opposite group  $\overline{M}$ , we change v's group label to the  $\overline{M}$ . Intuitively, this rule **filters the noise** from numerical approximation of spectral partitioning.

The Ensemble-GND and Ensemble-GNDR algorithm. We propose a variant of the standard GND(R) algorithm called Ensemble-GND and Ensemble-GNDR which consider the upper two issues in this section, that is, based on the standard GND algorithm [27], the Ensemble-GND(R) algorithm will (1) adjust the partition of the nodes in group M and  $\overline{M}$  according to their surrounding connectivity for specific target size (see details from the previous paragraphs of this section), and (2) select the result with the best performance in the ensemble with D (D = 1000 in this paper) results produced with D different initializations (instead of one initialization with the default seed, see details from the previous parts of this section).

Table 1. Properties of the networks we used in this paper.

	Crime	Petster-hamster	RoadEU	Political-blogs	Crime2	HI-II-14	DBLP
Nodes	754	2000	1177	1222	829	4165	12495
Links	2127	16714	1305	16714	1473	13087	49563

In this paper we use seven popular real world network datasets [41,38,36] to compare the performance of the existing algorithms and the proposed Ensemble-GND and Ensemble-GNDR algorithms. The properties of the networks are listed in Table 1. For all the dismantling tasks, we set the dismantling target size as the 1% of the original network size, i.e., when the GCC of the remaining network is smaller than the target size, the algorithm will stop to remove more nodes. In addition, for the unweighted cost case, the dismantling cost of any arbitrary nodes is the same. For the weighted cost case, the removal cost of any arbitrary nodes is equal to its remaining degree in the network. The running time of the one initialization of the GND algorithm are summarized in Table 2.

Table 2. The running time for one round of Ensemble-GND algorithm.

Running time(s)	Crime	Petster-hamster	RoadEU	Political-blogs	Crime2	HI-II-14	DBLP
Unweighted	0.147	1.200	0.193	1.184	0.146	1.047	23.551
Weighted case	0.407	3.798	0.374	4.151	0.426	4.730	53.399

The results of the Ensemble-GND(R) algorithm are listed in Table 3 and Table 4. In Table 3, we compared our Ensemble-GND(R) algorithm with the state-of-the-art algorithms, including BPD, Min-Sum, and GND(R) algorithms,

#### 8 Xiao-Long Ren, Nino Antulov-Fantulin

for weighted dismantling cost case and uniform cost (unweighted) case, respectively. We can clearly see that for all the weighted case and almost all the unweighted case (except the Political-blogs network), the Ensemble-GNDR can obtain the best performance.

**Table 3.** Comparison of the standard GND(R) and Ensemble-GND(R) algorithm by the dismantling cost. The better results are highlighted with bold text.

Unweighted case	BPD	Min-Sum	GND	Ensemble-GND	GNDR	Ensemble-GNDR
Crime Network	101	120	110	103	103	99
Petster Network	474	485	601	510	467	441
RoadEU Network	151	160	193	159	171	144
Political-blogs	<b>375</b>	380	494	435	404	386
Weighted case	BPD	Min-Sum	GND	Ensemble-GND	GNDR	Ensemble-GNDR
Weighted case Crime Network	BPD 0.594	Min-Sum 0.644	GND 0.642	Ensemble-GND 0.624	GNDR 0.584	Ensemble-GNDR 0.572
Weighted caseCrime NetworkPetster Network	BPD 0.594 0.829	Min-Sum 0.644 0.837	GND 0.642 0.914	Ensemble-GND 0.624 0.873	GNDR 0.584 0.810	Ensemble-GNDR 0.572 0.792
Weighted case Crime Network Petster Network RoadEU Network	BPD 0.594 0.829 0.463	Min-Sum 0.644 0.837 0.491	GND 0.642 0.914 0.523	Ensemble-GND 0.624 0.873 0.470	GNDR 0.584 0.810 0.464	Ensemble-GNDR 0.572 0.792 0.417

Further more, we also compared our algorithm with the deep reinforcement learning approach GraphDQN. The code of the GraphDQN method was not available at the time of writing of this paper. Therefore, we have made comparisons only on part of the networks, for which we had GraphDQN dismantling solutions (provided by the authors of study [36]). Based on all the five results in Table 4, we can conclude that the proposed Ensemble-GNDR algorithm has better performance than the deep reinforcement learning approach in both unweighted and weighted cost cases.

**Table 4.** Comparison of the Ensemble-GND(R) and the deep reinforcement learningbased algorithm GraphDQN. The dismantling results of the GraphDQN were obtained from the authors of the paper [36] (The code of the GraphDQN algorithm was not available at time we were writing this paper, but only the solutions on several datasets that we have used). The better results are highlighted with bold text.

Unweighted case	GraphDQN	Ensemble-GND	Ensemble-GNDR
Crime2 Network	185	183	161
HI-II-14 Network	553	483	412
DBLP Network	2496	2499	2064
Weighted case	GraphDQN	Ensemble-GND	Ensemble-GNDR
Crime2 Network	0.989	0.802	0.718
HI-II-14 Network	0.977	0.942	0.831

## 4 Conclusion

In this paper, we briefly reviewed the recent progress in the study of the network dismantling problem and explored the solution landscape of (generalized) network dismantling problem by proposing the Ensemble-GND and Ensemble-GNDR algorithm. We compared the proposed Ensemble-GND(R) algorithm with the state-of-the-art algorithms, including BPD, Min-Sum, and the standard GND(R) algorithms, as well as a recently proposed deep reinforcement learningbased algorithm GraphDQN. The results show that our Ensemble-GND(R) has a better performance both in the weighted case and the unweighted case of the network dismantling problem. Which opens new research directions of exploring the ensemble of dismantling solutions in the objective landscape by different methods of perturbations, initializations, and other modern machine learning optimizations techniques for highly non-linear and non-convex objective functions.

# 5 Acknowledgements

X.L.R. thanks to the financial support of China Scholarship Council (CSC). N.A.-F. thanks to the financial support from the EU Horizon 2020 project So-BigData under grant agreement No. 654024.

## References

- M. Fiedler, "Algebraic connectivity of graphs," *Czechoslovak Math J*, vol. 23, no. 2, pp. 298–305, 1973.
- R. Lipton, D. Rose, and R. Tarjan, "Generalized Nested Dissection," SIAM J Numer Anal, vol. 16, pp. 346–358, apr 1979.
- A. Pothen, H. D. Simon, and K.-P. Liou, "Partitioning sparse matrices with eigenvectors of graphs," SIAM J Matrix Anal Appl, vol. 11, pp. 430–452, jul 1990.
- T. N. Bui and C. Jones, "Finding good approximate vertex and edge partitions is NP-hard," *Information Processing Letters*, vol. 42, pp. 153–159, may 1992.
- S. Guattery and G. L. Miller, "On the quality of spectral separators," SIAM J Matrix Anal Appl, vol. 19, pp. 701–719, jul 1998.
- X.-L. Ren, N. Gleinig, D. Tolić, and N. Antulov-Fantulin, "Underestimated cost of targeted attacks on complex networks," *Complexity*, vol. 2018, pp. 1–15, 2018.
- R. Albert, H. Jeong, and A.-L. Barabási, "Error and attack tolerance of complex networks," *Nature*, vol. 406, no. 6794, pp. 378–382, 2000.
- R. Cohen, K. Erez, D. Ben-Avraham, and S. Havlin, "Breakdown of the Internet under intentional attack," *Phys Rev Lett*, vol. 86, no. 16, p. 3682, 2001.
- C. M. Schneider, A. A. Moreira, J. S. Andrade, S. Havlin, and H. J. Herrmann, "Mitigation of malicious attacks on networks," *Proc Natl Acad Sci USA*, vol. 108, no. 10, pp. 3838–3841, 2011.
- L. K. Gallos, R. Cohen, F. Liljeros, P. Argyrakis, A. Bunde, and S. Havlin, *Attack strategies on complex networks*, pp. 1048–1055. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006.

- 10 Xiao-Long Ren, Nino Antulov-Fantulin
- S.-M. Qin, X.-L. Ren, and L.-Y. L, "Efficient network dismantling via node explosive percolation," *Communications in Theoretical Physics*, vol. 71, p. 764, jun 2019.
- R. Pastor-Satorras and A. Vespignani, "Immunization of complex networks," *Phys Rev E*, vol. 65, no. 3, p. 36104, 2002.
- R. Pastor-Satorras, C. Castellano, P. Van Mieghem, and A. Vespignani, "Epidemic processes in complex networks," *Rev Mod Phys*, vol. 87, no. 3, pp. 925–979, 2015.
- N. Antulov-Fantulin, A. Lančić, T. Šmuc, H. Štefančić, and M. Šikić, "Identification of patient zero in static and temporal networks: Robustness and limitations," *Phys Rev Lett*, vol. 114, jun 2015.
- L. Lü, D. Chen, X.-L. Ren, Q.-M. Zhang, Y.-C. Zhang, and T. Zhou, "Vital nodes identification in complex networks," *Phys Rep*, vol. 650, pp. 1–63, sep 2016.
- D. Marx, "Parameterized graph separation problems," *Theoretical Computer Science*, vol. 351, pp. 394–406, feb 2006.
- T. Leighton and S. Rao, "Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms," *Journal of the ACM*, vol. 46, pp. 787– 832, nov 1999.
- S. Arora, S. Rao, and U. Vazirani, "Expander flows, geometric embeddings and graph partitioning," in *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing - STOC 04*, ACM Press, 2004.
- U. Feige, M. Hajiaghayi, and J. R. Lee, "Improved approximation algorithms for minimum weight vertex separators," *SIAM J Comput*, vol. 38, pp. 629–657, jan 2008.
- 20. S. Arora, E. Hazan, and S. Kale, "Fast algorithms for approximate semidefinite programming using the multiplicative weights update method," in 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS 05), IEEE, 2005.
- W. Ben-Ameur, M.-A. Mohamed-Sidi, and J. Neto, "The k-separator problem," in *Computing and Combinatorics* (D.-Z. Du and G. Zhang, eds.), (Berlin, Heidelberg), pp. 337–348, Springer Berlin Heidelberg, 2013.
- 22. S. Janson and A. Thomason, "Dismantling sparse random graphs," *Combin Probab Comput*, vol. 17, no. 2, pp. 259–264, 2008.
- Y. Deng and J. Wu, "Optimal attack strategy with heterogeneous costs in complex networks," in 2016 Annual IEEE Systems Conference (SysCon), pp. 1–5, April 2016.
- Y. Deng, J. Wu, Y. Xiao, M. Zhang, Y. Yu, and Y. Zhang, "Optimal disintegration strategy with heterogeneous costs in complex networks," *IEEE Transactions on* Systems, Man, and Cybernetics: Systems, pp. 1–9, 2018.
- G. Dong, J. Gao, R. Du, L. Tian, H. E. Stanley, and S. Havlin, "Robustness of network of networks under targeted attack," *Phys Rev E*, vol. 87, p. 052804, May 2013.
- A. Braunstein, L. Dall'Asta, G. Semerjian, and L. Zdeborová, "Network dismantling," Proc Natl Acad Sci USA, vol. 113, pp. 12368–12373, nov 2016.
- X.-L. Ren, N. Gleinig, D. Helbing, and N. Antulov-Fantulin, "Generalized network dismantling," *Proceedings of the National Academy of Sciences*, vol. 116, no. 14, pp. 6554–6559, 2019.
- I. Dinur and S. Safra, "On the hardness of approximating minimum vertex cover," Annals of Mathematics, vol. 162, p. 2005, 2004.
- 29. S. Becker, Y. Zhang, and A. A. Lee, "Geometry of energy landscapes and the optimizability of deep neural networks," 2018.

- J. Calatayud, R. Bernardo-Madrid, M. Neuman, A. Rojas, and M. Rosvall, "Exploring the solution landscape enables more reliable network community detection," 2019.
- 31. B. H. Good, Y.-A. de Montjoye, and A. Clauset, "Performance of modularity maximization in practical contexts," *Phys. Rev. E*, vol. 81, p. 046106, Apr 2010.
- 32. S. Mugisha and H.-J. Zhou, "Identifying optimal targets of network attack by belief propagation," *Phys Rev E*, vol. 94, p. 012305, Jul 2016.
- H.-J. Zhou, "Spin glass approach to the feedback vertex set problem," Eur Phys J B, vol. 86, p. 455, Nov 2013.
- F. Morone and H. A. Makse, "Influence maximization in complex networks through optimal percolation," *Nature*, vol. 524, no. 7563, p. 65, 2015.
- R. Bar-Yehuda and S. Even, "A linear-time approximation algorithm for the weighted vertex cover problem," J Algorithms, vol. 2, no. 2, pp. 198 – 203, 1981.
- C. Fan, Y. Sun, Z. Li, Y.-Y. Liu, M. Chen, and Z. Liu, "Dismantle large networks through deep reinforcement learning," *ICLR 2019*, 2019.
- Y. Chen, G. Paul, S. Havlin, F. Liljeros, and H. E. Stanley, "Finding a better immunization strategy," *Phys Rev Lett*, vol. 101, p. 58701, jul 2008.
- L. Zdeborová, P. Zhang, and H.-J. Zhou, "Fast and simple decycling and dismantling of networks," *Sci Rep*, vol. 6, p. 37954, 2016.
- A. Buluç, H. Meyerhenke, I. Safro, P. Sanders, and C. Schulz, *Recent Advances in Graph Partitioning*, pp. 117–158. Cham: Springer International Publishing, 2016.
- M. Matsumoto and T. Nishimura, "Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator," ACM Trans. Model. Comput. Simul., vol. 8, pp. 3–30, Jan. 1998.
- J. Kunegis, "The koblenz network collection," in Proc Int Web Observatory Workshop, pp. 1343–1350, 2013.