The Combined Critical Node and Edge Detection Problem. An Evolutionary Approach

Tamás Képes, Noémi Gaskó(✉), and Géza Vekov

Babeş-Bolyai University, Cluj-Napoca, Romania
{tamas.kepes,noemi.gasko,geza.vekov}@ubbcluj.ro

Abstract. Studying complex networks has received a great deal of attention in recent years. A relevant problem is detecting critical nodes - nodes which, based on some measures, are more important than others in a certain network. In this paper, we propose a new optimization problem: the critical node and edge detection problem, which combines two well-known problems. A simple genetic algorithm is proposed to solve this problem, with numerical experiments having shown the potential of the method. As an application, we analyze several real-world networks and use the introduced problem as a new network robustness measure.

Keywords: Critical nodes · Critical edges · Genetic algorithm · Complex networks

1 Introduction

The study of complex networks has gained increased attention in recent years due to its applicability in different research fields (e.g. biology [1], ecology [7], telecommunication [12]). A relevant problem in networks study is the identification of a set of nodes, which, based on some properties, can be considered more important than others. If this property is a network measure, the problem is called the critical node detection problem.

The critical node detection problem (CNDP) has several application possibilities in different research fields, e.g. social network analysis [8,14], network risk management [5] and network vulnerability studies [10].

Generally, the CNDP consists of finding a set of k nodes in a given graph $G = (V, E)$, which, if deleted, maximally degrades the graph according to a given measure σ (σ can be for example, betweenness centrality, closeness centrality or page rank [18,24]).

This work was supported by a grant of the Romanian National Authority for Scientific Research and Innovation, CNCS - UEFISCDI, project number PN-III-P1-1.1-TE-2019-1633.

G. Rudolph et al. (Eds.): PPSN 2022, LNCS 13398, pp. 324–338, 2022.
https://doi.org/10.1007/978-3-031-14714-2_23

The definition of critical edge detection is almost the same. Given a graph $G = (V, E)$, the goal is to find a set of l edges in order to optimize a certain network property.

We propose a new combinatorial optimization problem, the critical node and edge detection problem. Although the critical node detection and the critical edge detection problems exist separately, the unification of the two problems is essential in some applications (e.g. road networks, computer networks, etc.) as it can model real-world situations better. In a certain network not only nodes but also edges can be deleted.

The next section of the paper presents related work about critical node and edge detection variants and algorithms. The third section describes the new proposed critical node and edge detection problem. In the fourth section, the designed algorithms are described, while section five presents the numerical results. The article ends with conclusions and further work.

2 Related Work

The CNDP can be considered as a special case of the node deletion problem [22]. Based on the survey [21] the CNDP can be divided in two main classes. The first class is the k-vertex-CNDP, where in the given graph $G = (V, E)$ and a connectivity metric σ and a given number k and the goal is to minimize the objective function $f(\sigma)$ of deleting k nodes. Some problems from this class are the MaxNum problem [32] (maximizing the number of connected components), MinMaxC (minimizing the size of the largest components) [33], and CNP (critical node problem - minimizing pairwise connectivity) [26]. The most studied variant is the CNP, with several algorithm proposals, for example, using integer linear programming [4], iterated local search algorithms [23], and greedy randomized adaptive search procedures [29]. In [3] an evolutionary framework is proposed which can deal with several variants of the CNDP.

The other main class is the β-connectivity-CNDP, where for a given $G = (V, E)$ graph, a connectivity metric σ and an integer β, the main goal is to limit the objective function $f(\sigma)$ to β, while minimizing the number of deleted nodes. Examples from this class are the Cardinality-Constrained-CNP (CC-CNP) [6] or the Component-Cardinality-Constrained CNP (3C-CNP) [20].

In an early work [40], edge deletion was studied in the case of maximum flow networks. In [37], the edge interdiction clique problem is introduced, where edges need to be removed so that the size of the maximum clique in the remaining graph is minimized and an exact algorithm is proposed to solve it for small graphs. In [15], a branch-and-cut algorithm is proposed to solve the problem.

In the matching interdiction problem [43] the weight of the maximum matching in the remaining graph after deleting edges or nodes needs to be minimized. In the same article, a pseudo-polynomial algorithm is proposed.

A recent article [9] proposes the online node and edge detection problem, where there are discussed and analyzed some online edge and node deletion problems. In [27] vertex and edge protection is proposed to stop a spreading process.

Simultaneous deletion of nodes and links, for the best of our knowledge, appeared only in two variants: in [39] a joint region identification is proposed and in [11] the β-connectivity CNDP is studied where both nodes and edges can be deleted.

3 Combined Critical Node and Edge Detection Problem

The critical node and edge detection problem (CNEDP) consists of finding a set $W \subseteq V$ containing k nodes and a set $F \subseteq E$ having l edges in a given graph $G = (V, E)$, which deleted maximally degrades the graph according to a given measure σ. We denote this introduced problem by (k, l)-CNEDP.

In this article we study as a network connectivity measure the pairwise connectivity. The objective function, which needed to be minimized is the following:

$$f(A) = \sum_{C_i \in G[V \setminus A]} \frac{\delta_i(\delta_i - 1)}{2}, \tag{1}$$

where $A \subseteq V$, C_i is the set of connected components in the remaining graph, after the deletion of nodes and edges, and δ_i is the size of the connected component C_i.

Remark 1. It is obvious that $(k, 0)$-CNEDP reduces the CNDP, and $(0, l)$-CNEDP reduces the critical edge detection problem (CEDP).

Example 1. Let us consider the graph presented in Fig. 1. Considering (1,1)-CNDEP, if deleting the sixth node and the edge between node 3 and 4, $A = \{1, 2, 3, 4, 5, 7\}$, $C_1 = \{1, 2, 3\}$, $C_2 = \{4\}$, $C_3 = \{5, 7\}$, $\delta_1 = \{3\}$, $\delta_2 = \{1\}$, $\delta_3 = \{2\}$,

$$f(A) = \frac{3(3-1)}{2} + \frac{1(1-1)}{2} + \frac{2(2-1)}{2} = 4$$

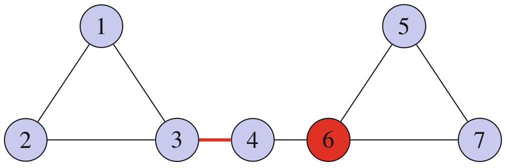

Fig. 1. A simple graph with 7 nodes

Remark 2. The complexity of the (k, l)-CNEDP is NP-complete. In [4] it is proved that the decision variant of the CNP problem is NP-complete, CNP is a subtask of the (k, l)-CNDEP problem.

4 Methods

We propose two algorithms to solve the (k, l)-CNEDP presented in the following.

4.1 Greedy Algorithm

In the framework proposed in [3] three non-evolutionary greedy solutions are presented to solve the CNDP. We adapted the second algorithm to fit the CNEDP, using the pairwise connectivity in the greedy decision-making process. The greedy selection is made based on the following function:

$$GR2(S_X) = argmax\{(f(S_X) - f(S_X \cup \{t\}) : t \in X \setminus S_X\}$$

where S_X is one of S_{nodes} or S_{edges}, and X represents respectively the original set of nodes or edges of the network.

The algorithm selects the best nodes and edges that minimize the objective function and, depending on the values of k and l, selects a single node or edge randomly, which will be removed from the network. The procedure is repeated until the maximal number of removed nodes and edges is reached. Since the selection is made randomly from the pre-selected components that have maximal impact on pairwise connectivity, the algorithm should be repeated several times to achieve the best possible results. According to [3] the number of iterations to perform should be set to $|S_{nodes}|^2$, thus we can achieve a feasible solution by setting this value to $(|S_{nodes}| + |S_{edges}|)^2$. The original framework recommends the execution of the other greedy selection rule to minimize the removed network component count after reaching a solution.

$$GR1(S_X) = argmax\{(f(S_X \setminus t) - f(S_X) : t \in S_X\}$$

Since the pairwise connectivity is driven by both the removed nodes and edges, and because of the specific structure of $(k, l) - CNEDP$, we consider that this step is not mandatory. We can argue that the complexity of the reintroduction of the removed components is too resource-intensive, but in case it is required, it can be executed. The outline of the algorithm is presented in Algorithm 1.

The number of total fitness function execution can be approximated for the algorithm, since the method is set to stop when k edges and l nodes are reached in the removal process. In each iteration the $GR2$ method computes the fitness of the network if any of the remaining nodes and edges are removed, one at a time, selecting the $Best*$ items those resulting in the best fitness value. Let us suppose that $D_{average}$ is the average node degree, and that with the removal of every node the number of fitness calculations in the next iteration will decrease with $1 + D_{average}$, and with the removal of an edge it decreases with 1. Then the approximate number of fitness execution will be $l(V + E) + \frac{1}{2}l(l + 1)(1 + D_{average}) + \frac{1}{2}k(2V - k)$.

Algorithm 1. Greedy algorithm

Parameters:

- G the network
- k the number of edges to remove
- l the number of nodes to remove
- S_{edges} and S_{nodes}, the set of nodes and edges to remove.
- $GR2(S_X, 1, a)$ - greedy selection algorithm notation based on [3]

```
$S_{edges} = \{\emptyset\}$
$S_{nodes} = \{\emptyset\}$
**while** $(|S_{edges}| < l)$ **or** $(|S_{nodes}| < k)$ **do**
    $Best_{edges} = \{GR2(S_{edges}, 1, a)\}$
    $Best_{nodes} = \{GR2(S_{nodes}, 1, a)\}$
    $[S_{nodes}, S_{edges}] = [S_{nodes}, S_{edges}] \cup Select(Best_{edges}, Best_{nodes})$
**end while**
**return** $S_{nodes}, S_{edges}$
```

4.2 Genetic Algorithm

We designed a simple genetic algorithm, the encoding, fitness evaluation and the operators used are presented in the next. The outline of the genetic algorithm is described in Algorithm 2.

Encoding: An individual is represented by two lists, one for nodes and the other for edges.

Fitness: Each individual is evaluated according to the pairwise connectivity of the connected components in the network, after the removal of the individual's nodes and edges.

Crossover and Parent Selection: This is realized using a tournament-based selection. For each round of the crossover tournament, the algorithm randomly chooses a set number of individuals from the population after which a selection is made, keeping only the two best individuals according to their fitness. They will then reproduce, by combining their node and edge lists. The algorithm will then split these lists randomly and evenly, keeping some restrictions, such as uniqueness. This way we generate two children for each round of the tournament. At the end of the crossover tournament, a set of new individuals is created (Algorithm 3).

Mutation: Two types of mutation are used. The first one is done by randomly replacing either a node or an edge in the offspring. The chance of either selection is 50%. Our new node or edge selection takes into account uniqueness inside an individual.

The second mutation is a time-varying operator. In the first step 50% of the $k + l$ nodes and edges are changed. The number of changes decreases linearly until half of the maximum generation number is reached, after which it will equal one.

While we have strict restrictions on each individual in our population, a repair operator is not necessary, since both the crossover and mutation operators self-repairs the potentially caused damage to any given individual.

Selection of Survivors: The algorithm combines the original population and any newly-created child, including the possible mutations, into a larger set of individuals, after which we trim this new set to the original population size using elitism (keeping the best individuals), this will become the new population for the next iteration of the algorithm (a $(\mu + \lambda)$ selection scheme is used).

Algorithm 2. Genetic algorithm

Parameters:

- G the network
- *pop_size* the number of individuals in the population
- k and l, the number of nodes and edges in an individual.
- p_{mut} the chance of mutation

```
Randomly initialize pop;
repeat
  Evaluate current population based on fitness value;
  Create child population using tournament based crossover;
  if random chance == $p_{mut}$ then
    Choose a random child from list of children.
    Mutate child by randomly replacing either a node or an edge with a new one;
  end if
  Elitist selection of pop_size number of individuals from combined parent and children population;
until Maximum number of generations;
return Best individual from final pop;
```

4.3 Experimental Set-Up

Benchmarks The synthetic benchmark set proposed in [38] contains three types of graphs, with different basic properties: Barabási-Albert (BA) - scale-free networks, Erdős-Rényi (ER) - random networks, and Forest-fire (FF) graphs, which simulate how fire spreads through a forest. Table 1 describes basic network measures of the benchmarks employed: number of nodes ($|V|$), number of edges ($|E|$), average degree ($\langle d \rangle$), density of the graph (ρ), and average path length (l_G).

In Table 2 the set of real networks used for numerical experiments is presented, including the source of the network.

Algorithm 3. Parent selection and Crossover

Parameters:

- *pop* the current population in the genetic algorithm.
- *tournament_size* the number of selected individuals to partake in the tournament.
- *max_round_number* the maximum number of rounds for the tournament, equal to half the size of newly generated child population

```
repeat
    Select tournament_size number of individuals to participate;
    Select the two best individuals according to evaluation from tournament contenders;
    Unite and then split evenly the two parents' node and edge lists.
    Append new children to the result children population.
until Maximum number of tournament rounds;
return child_pop
```

Table 1. Synthetic benchmark test graphs and basic properties.

Graph	$\|V\|$	$\|E\|$	$\langle d \rangle$	ρ	l_G
BA500	500	499	1.996	0.004	5.663
BA1000	1000	999	1.998	0.002	6.045
ER250	235	350	2.979	0.013	5.338
ER500	466	700	3.004	0.006	5.973
FF250	250	514	4.112	0.017	4.816
FF500	500	828	3.312	0.007	6.026

Parameter Setting. To find a good parameter configuration 16 parameter settings were tested on four networks: two synthetic (ER250 and FF250) and two real world networks (dolphins and karate). Table 3 presents the tested parameter settings, and Fig. 2 presents the obtained results. Based on a Wilcoxon non-parametric statistical test the configuration S11 was chosen for the further experiments.

The number of critical nodes (k) is 5% of the total nodes, while the number of critical edges (l) is set to 3% of the total number of edges (proportions are set general, to emphasize critical nodes and edges on different type of networks). The maximum generation number for both GA variants was set to 5000.

4.4 Results and Discussion

An example of the evolution of the fitness value of the genetic algorithm is presented in Fig. 3, we can observe the change of the values in each step.

For better understanding, Fig. 4 presents the smallest network, the Zebra network, and critical nodes and edges detected with the genetic algorithm.

Table 4 presents the results obtained from the genetic algorithm (GA_1), genetic algorithm with time-varying mutation (GA_2) and from the greedy algo-

Table 2. Real graphs and basic properties.

Graph	$\|V\|$	$\|E\|$	$\langle d\rangle$	ρ	l_G	Ref
Bovine	121	190	3.140	0.026	2.861	[30]
Circuit	252	399	3.167	0.012	5.806	[28]
Dolphins	62	159	5.1290	0.0841	3.3570	[19]
Ecoli	328	456	2.780	0.008	4.834	[25, 41]
Football	115	613	10.6609	0.0935	2.5082	[16, 19]
Hamsterster	2426	16631	13.7106	0.0057	2.4392	[31]
HumanDis	516	1188	4.605	0.008	6.509	[17]
Karate	34	78	4.5882	0.1390	2.4082	[19, 42]
Zebra	27	111	8.2222	0.3162	1.3590	[19, 36]

Table 3. Parameter setting used for parameter tuning

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	15	S16
Pop. size	50	50	50	50	50	50	50	50	100	100	100	100	100	100	100	100
p_c	0.8	0.9	0.8	0.9	0.8	0.9	0.8	0.9	0.8	0.9	0.8	0.9	0.8	0.9	0.8	0.9
p_m	0.02	0.02	0.05	0.05	0.02	0.02	0.05	0.05	0.02	0.02	0.05	0.05	0.02	0.02	0.05	0.05
Tournament s.	3	3	3	3	5	5	5	5	3	3	3	3	5	5	5	5

rithm. GA_1 and GA_2 outperformed the greedy algorithm in most cases. Only in the case of the Football network did both algorithms perform in the same way (with standard deviation equal to 0). However, analyzing the results, in the case of the Football network the values for k and l were too small, because there was no change from the initial population in GA_1 and GA_2. The incorporation of time-varying mutation did not significantly improve the results.

Regarding the running time of both methods (GA_1 and greedy), in small networks, as expected, greedy runs faster (e.g. in the case of dolphins network 2.47 ± 0.02 s running time has the greedy algorithm, and 183.64 ± 0.48 s the GA_1), but in a larger network the GA_1 has better running time (e.g. for the FF500 network the greedy runs in average 1420.66 ± 63.48 s and the GA_1 1294.3 ± 25.15 s).

4.5 Application: New Network Robustness Measure Proposal

As an application of critical node and edge detection, we introduce a new network robustness measure. In the literature several robustness measure exist, trying to capture different properties of the networks. For example [13] describes different measures to characterize network robustness: k_v - vertex connectivity - the minimal number of vertices which need to be removed to disconnect the graph, k_e- edge connectivity - the same measure for edges, diameter of the graph (d), average distance ($d-$), average efficiency (E) - considering shortest paths, maximum edge betweenness (b_em), average vertex betweenness (b_v), average edge

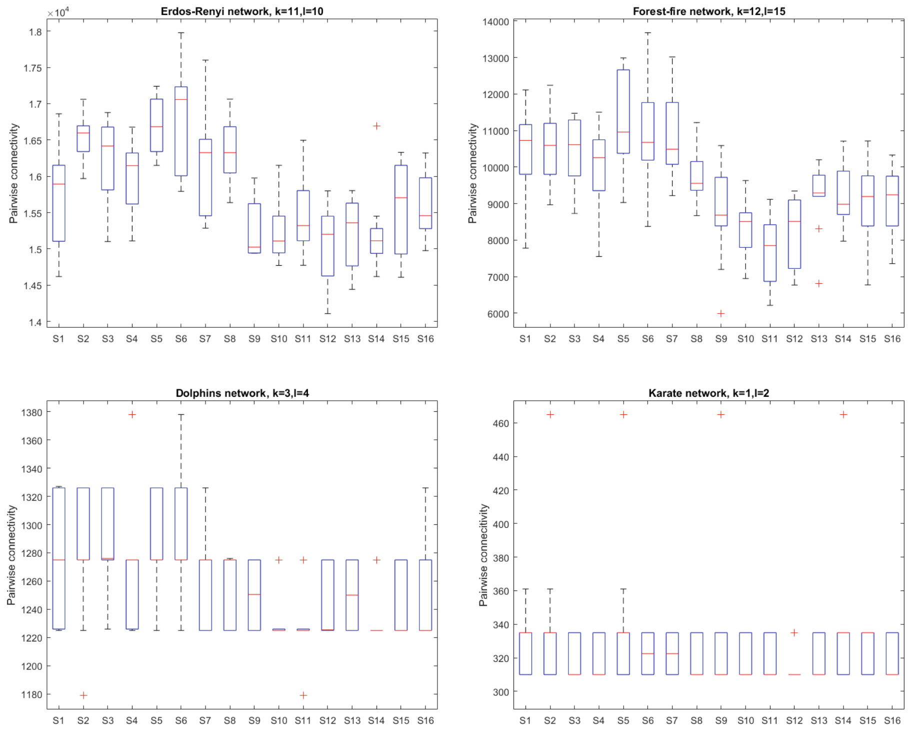

Fig. 2. Pairwise connectivity values over ten independent runs for four networks for 16 different parameter configurations

betweenness (b_e) - these measures considering shortest paths. The average clustering coefficient (C) is a proportion of triangles and connected triples. Algebraic connectivity (λ_2) is the second smallest eigenvalue of the Laplacian matrix of G, a number of spanning trees (ϵ) counts the possible different spanning trees of the graph, while effective graph resistance (Kirchhoff index) (R) investigates the graph as a network circuit.

We study several real-world networks from different application fields: two infrastructure networks - UsAir97 [31] a weighted undirected network with 332 nodes and 2126 edges (we do not take into account the weights) containing flights in the US in the year 1997 (nodes are airports and edges represent direct flight between them) and a road network [19,35] containing international E-roads, nodes representing cities (1174 nodes) and edges representing direct E-road connections between them (1417 edges). Two brain networks are studied: a mouse visual cortex network [2] with 123 nodes and 214 edges, and a cat-mixed-brain-species-1 network [2] with 65 nodes and 1100 edges (we will use the abbreviations Mouse_cortex and Cat_brain in the next). Two power networks are studied: 494-bus [31] (494 nodes and 586 edges) and 662-bus [31] (662 nodes and 906 edges), two interaction networks (Infect-dublin [34] having 410 nodes

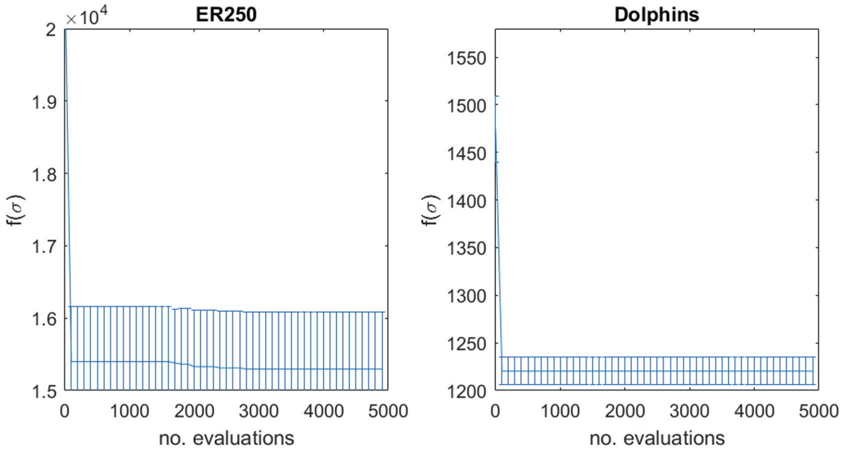

Fig. 3. Errorbar of 10 independent runs representing fitness values over evaluations on two example graphs: an Erdos-Renyi graph and Dolphins network

and 2800 edges and Infect-hyper [34] with 113 nodes and 2200 edges), and a computer network - Route_views [19] (6474 nodes and 13895 edges).

The above mentioned network measures are calculated for the studied networks, as presented in Table 5. As we can see, the majority of the indices cannot be used for disconnected networks (in this example, the E-road network is disconnected), this is one of the motivations to introduce the new measure to analyze the network robustness, based on the (k,l)-CNEDP.

The introduced measure ($NE_{k,l}$) has the following form:

$$NE_{k,l} = \frac{2 \cdot (k,l)\text{-CNEDP}}{(n-k-1)(n-k-2)}$$

$\frac{(n-k-1)(n-k-2)}{2}$ is the worst possible value of pairwise connectivity, after deleting k nodes, n is the number of nodes in the original network, $NE_{k,l} \in [0,1]$.

In the case of the USAir97 network, for example:

$$NE_{21,6} = \frac{2 \cdot 35264}{(332-21-1)(332-21-2)} = 0.66.$$

The $NE_{k,l}$ can be seen as a measure which based on the number of deletion of nodes and edges quantifies the network robustness. To analyse the results a correlation matrix was built (without the results of the E-road network), the new measure - $NE_{k,l}$ (new m) was compared with d, $d-$, E, b_em, b_v and C. As presented in Fig. 5 a weak correlation exists between the new measure and the clustering coefficient (C).

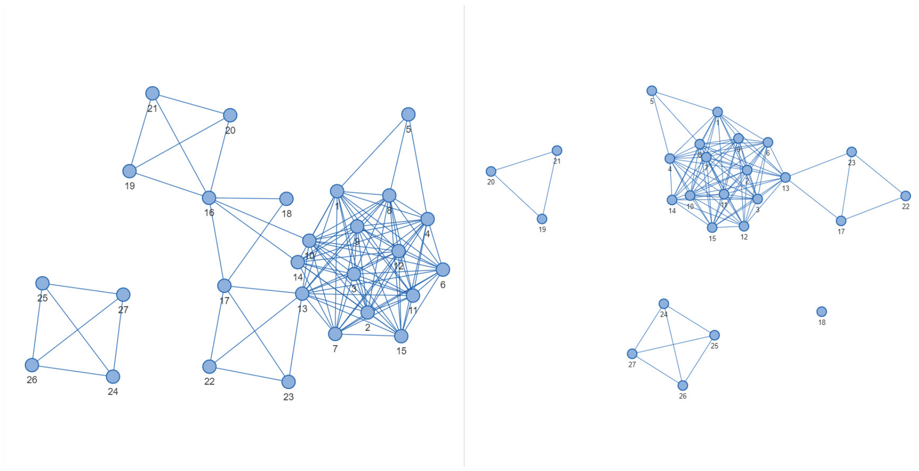

Fig. 4. The smallest real-world network, the zebra network (left). The remaining network after node and edge deletion - after (1, 3)-CNEDP (right)

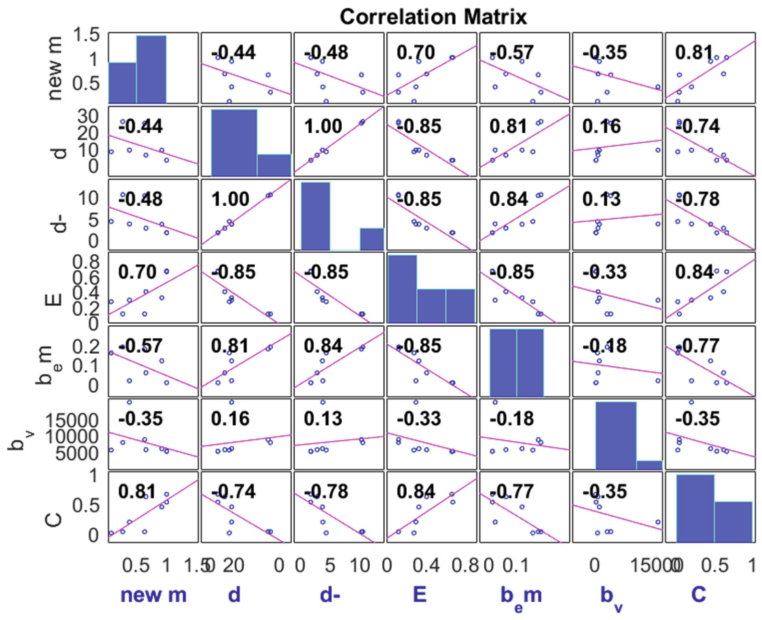

Fig. 5. Pearson's correlation coefficients between network robustness mesures

Table 4. Results for synthetic and real graphs. Mean values and standard deviation over 10 independent runs are presented. A (*) indicates best result based on a Wilcoxon sign-rank test

Graph	GA_1	GA_2	Greedy
BA500	$650.70 \pm 82.62^*$	835.00 ± 259.57	5722.90 ± 33.20
BA1000	$1947.20 \pm 235.93^*$	$2021.10 \pm 255.62^*$	362555.90 ± 924.58
ER250	$15296.30 \pm 785.33^*$	$14976.70 \pm 623.52^*$	25066.70 ± 156.40
ER500	$61664.60 \pm 1532.08^*$	$62929.20 \pm 2215.60^*$	100357.40 ± 568.82
FF250	$7905.50 \pm 1212.81^*$	$7909.70 \pm 619.53^*$	25508.20 ± 1052.20
FF500	$12303.30 \pm 4858.39^*$	$12342.90 \pm 4493.16^*$	105350.50 ± 4396.26
Bovine	$16.50 \pm 20.66^*$	$29.20 \pm 25.50^*$	1337.10 ± 31.78
Circuit	$274.00 \pm 332.22^*$	$274.70 \pm 417.62^*$	24110.10 ± 1448.70
Dolphins	$1220.40 \pm 14.55^*$	$1230.20 \pm 15.75^*$	1711.00 ± 0.00
Ecoli	$447.60 \pm 465.91^*$	$1338.40 \pm 1467.07^*$	47649.30 ± 576.69
Football	$5995.00 \pm 0.00^*$	$5995.00 \pm 0.00^*$	$5995.00 \pm 0.00^*$
HumanDis	$531.70 \pm 859.40^*$	3927.30 ± 9258.98	113625.90 ± 1521.87
Karate	$355.90 \pm 41.98^*$	$315.00 \pm 10.54^*$	411.10 ± 80.67
Zebra	$165.20 \pm 7.32^*$	$166.40 \pm 8.98^*$	237.00 ± 0.00

Table 5. Network robustness measures for studied networks

Measure	Networks								
	USAir97	E-road	Mouse_cortex	Cat_brain	494-bus	662-bus	Infect-dublin	Infect-hyper	Route_views
k_v	1	0	1	3	1	1	1	1	1
k_e	1	0	1	3	1	1	1	1	1
d	6	∞	8	3	26	25	9	3	9
$d-$	2.73	∞	4.27	1.69	10.47	10.24	3.63	1.65	3.70
E	0.40	0	0.27	0.66	0.11	0.11	0.32	0.67	0.29
$b_e m$	0.06	–	0.16	0.01	0.19	0.18	0.12	0.01	0.02
b_v	618.65	–	506.01	86.38	2827.38	3716.30	947	148.75	15227.74
b_e	70.76	–	369.78	4.84	1180.51	1429.46	110.10	4.77	5586.99
C	0.62	0	0.02	0.66	0.04	0.04	0.45	0.53	0.25
λ_2	0.12	0	0.03	2.88	−4.75	−5.14	0.19	0.99	−25.84
ϵ	3.37e+234	0	6.41e+10	1.45e+81	0	1.004	∞	1.7e+169	∞
R	45538.19	∞	47450.21	262.08	−4.56e+18	1.78e+18	30563.17	527.78	1587664.19
$NE^{a}_{k,l}$	0.66	0.33	0.09	1.00	0.28	0.64	0.92	1	0.39

[a] k and l are chosen as 3% of the nodes and 1% of edges

5 Conclusions

A new combinatorial optimization problem, the combined CNEDP, is defined and analyzed. Two methods are proposed to solve this problem: a greedy approach and a simple GA. Numerical experiments on both synthetic and real-world networks show the effectiveness of the proposed algorithm. As a direct application this newly-introduced problem is used as a new network centrality measure for network robustness testing. Further work will address other network measures (for example maximum components size, network centrality measures) and the refinement of the GA.

References

1. Albert, R.: Scale-free networks in cell biology. J. Cell Sci. **118**(21), 4947–4957 (2005)
2. Amunts, K., et al.: Bigbrain: an ultrahigh-resolution 3d human brain model. Science **340**(6139), 1472–1475 (2013)
3. Aringhieri, R., Grosso, A., Hosteins, P., Scatamacchia, R.: A general evolutionary framework for different classes of critical node problems. Eng. Appl. Artif. Intell. **55**, 128–145 (2016)
4. Arulselvan, A., Commander, C.W., Elefteriadou, L., Pardalos, P.M.: Detecting critical nodes in sparse graphs. Comput. Oper. Res. **36**(7), 2193–2200 (2009)
5. Arulselvan, A., Commander, C.W., Pardalos, P.M., Shylo, O.: Managing network risk via critical node identification. Risk management in telecommunication networks. Springer (2007)
6. Arulselvan, A., Commander, C.W., Shylo, O., Pardalos, P.M.: Cardinality-constrained critical node detection problem. In: Gülpınar, N., Harrison, P., Rüstem, B. (eds.) Performance Models and Risk Management in Communications Systems. SOIA, vol. 46, pp. 79–91. Springer, New York (2011). https://doi.org/10.1007/978-1-4419-0534-5_4
7. Bascompte, J.: Networks in ecology. Basic Appl. Ecol. **8**(6), 485–490 (2007)
8. Borgatti, S.P.: Identifying sets of key players in a social network. Comput. Math. Organ. Theory **12**(1), 21–34 (2006)
9. Chen, L.H., Hung, L.J., Lotze, H., Rossmanith, P.: Online node-and edge-deletion problems with advice. Algorithmica **83**(9), 2719–2753 (2021)
10. Dinh, T.N., Thai, M.T.: Precise structural vulnerability assessment via mathematical programming. In: 2011-MILCOM 2011 Military Communications Conference, pp. 1351–1356. IEEE (2011)
11. Dinh, T.N., Thai, M.T.: Network under joint node and link attacks: Vulnerability assessment methods and analysis. IEEE/ACM Trans. Networking **23**(3), 1001–1011 (2014)
12. Dzaferagic, M., Kaminski, N., McBride, N., Macaluso, I., Marchetti, N.: A functional complexity framework for the analysis of telecommunication networks. J. Complex Networks **6**(6), 971–988 (2018)
13. Ellens, W., Kooij, R.E.: Graph measures and network robustness. arXiv preprint arXiv:1311.5064 (2013)
14. Fan, N., Pardalos, P.M.: Robust optimization of graph partitioning and critical node detection in analyzing networks. In: Wu, W., Daescu, O. (eds.) COCOA 2010. LNCS, vol. 6508, pp. 170–183. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17458-2_15
15. Furini, F., Ljubić, I., San Segundo, P., Zhao, Y.: A branch-and-cut algorithm for the edge interdiction clique problem. Eur. J. Oper. Res. **294**(1), 54–69 (2021)
16. Girvan, M., Newman, M.E.: Community structure in social and biological networks. Proc. Natl. Acad. Sci. **99**(12), 7821–7826 (2002)
17. Goh, K.I., Cusick, M.E., Valle, D., Childs, B., Vidal, M., Barabási, A.L.: The human disease network. Proc. Natl. Acad. Sci. **104**(21), 8685–8690 (2007)
18. Iyer, S., Killingback, T., Sundaram, B., Wang, Z.: Attack robustness and centrality of complex networks. PLoS ONE **8**(4), e59613 (2013)
19. Kunegis, J.: Konect: The koblenz network collection. In: Proceedings of the 22nd International Conference on World Wide Web, WWW 2013, pp. 1343–1350. Companion, Association for Computing Machinery, New York (2013). https://doi.org/10.1145/2487788.2488173

20. Lalou, M., Tahraoui, M.A., Kheddouci, H.: Component-cardinality-constrained critical node problem in graphs. Discret. Appl. Math. **210**, 150–163 (2016)
21. Lalou, M., Tahraoui, M.A., Kheddouci, H.: The critical node detection problem in networks: a survey. Comput. Sci. Rev. **28**, 92–117 (2018)
22. Lewis, J.M., Yannakakis, M.: The node-deletion problem for hereditary properties is np-complete. J. Comput. Syst. Sci. **20**(2), 219–230 (1980)
23. Lourenço, H.R., Martin, O.C., Stützle, T.: Iterated local search: framework and applications. In: Gendreau, M., Potvin, J.-Y. (eds.) Handbook of Metaheuristics. ISORMS, vol. 272, pp. 129–168. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-91086-4_5
24. Lozano, M., García-Martínez, C., Rodriguez, F.J., Trujillo, H.M.: Optimizing network attacks by artificial bee colony. Inf. Sci. **377**, 30–50 (2017)
25. Lusseau, D., Schneider, K., Boisseau, O.J., Haase, P., Slooten, E., Dawson, S.M.: The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations. Behav. Ecol. Sociobiol. **54**(4), 396–405 (2003)
26. Marx, D.: Parameterized graph separation problems. Theoret. Comput. Sci. **351**(3), 394–406 (2006)
27. Michalak, K.: Evolutionary graph-based V+E optimization for protection against epidemics. In: Bäck, T., Preuss, M., Deutz, A., Wang, H., Doerr, C., Emmerich, M., Trautmann, H. (eds.) PPSN 2020. LNCS, vol. 12270, pp. 399–412. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58115-2_28
28. Milo, R., et al.: Superfamilies of evolved and designed networks. Science **303**(5663), 1538–1542 (2004)
29. Purevsuren, D., Cui, G., Win, N.N.H., Wang, X.: Heuristic algorithm for identifying critical nodes in graphs. Adv. Comput. Sci. Int. J. **5**(3), 1–4 (2016)
30. Reimand, J., Tooming, L., Peterson, H., Adler, P., Vilo, J.: Graphweb: mining heterogeneous biological networks for gene modules with functional significance. Nucleic Acids Res. **36**, 452–459 (2008)
31. Rossi, R.A., Ahmed, N.K.: The network data repository with interactive graph analytics and visualization. In: AAAI (2015)
32. Shen, S., Smith, J.C.: Polynomial-time algorithms for solving a class of critical node problems on trees and series-parallel graphs. Networks **60**(2), 103–119 (2012)
33. Shen, S., Smith, J.C., Goli, R.: Exact interdiction models and algorithms for disconnecting networks via node deletions. Discret. Optim. **9**(3), 172–188 (2012)
34. SocioPatterns: Infectious contact networks. http://www.sociopatterns.org/datasets/
35. Šubelj, L., Bajec, M.: Robust network community detection using balanced propagation. The European Physical Journal B **81**(3), 353–362 (2011)
36. Sundaresan, S.R., Fischhoff, I.R., Dushoff, J., Rubenstein, D.I.: Network metrics reveal differences in social organization between two fission-fusion species, grevy's zebra and onager. Oecologia **151**(1), 140–149 (2007)
37. Tang, Y., Richard, J.P.P., Smith, J.C.: A class of algorithms for mixed-integer bilevel min-max optimization. J. Global Optim. **66**(2), 225–262 (2016)
38. Ventresca, M.: Global search algorithms using a combinatorial unranking-based problem representation for the critical node detection problem. Comput. Oper. Res. **39**(11), 2763–2775 (2012)
39. Wang, S., Zhang, T., Feng, C.: Nodes and links jointed critical region identification based network vulnerability assessing. In: 2016 IEEE International Conference on Network Infrastructure and Digital Content (IC-NIDC), pp. 66–71 (2016). https://doi.org/10.1109/ICNIDC.2016.7974537

40. Wollmer, R.: Removing arcs from a network. Oper. Res. **12**(6), 934–940 (1964)
41. Yang, R., Huang, L., Lai, Y.C.: Selectivity-based spreading dynamics on complex networks. Phys. Rev. E **78**(2), 026111 (2008)
42. Zachary, W.W.: An information flow model for conflict and fission in small groups. J. Anthropol. Res. **33**(4), 452–473 (1977)
43. Zenklusen, R.: Matching interdiction. Discret. Appl. Math. **158**(15), 1676–1690 (2010)