Research report

GT-NDNetw: Abordări bazate pe Teoria Jocurilor pentru Problema Detectării Nodurilor Critice în Rețele Sociale și Economice

PN-III-P1-1.1-TE-2019-1633, 56/2020

Phase 2

Objective 1. Extension of the CND problem to hypergraphs and multilayer networks (first part)

A 2.1. Documentation of existing connectivity metrics for hypergraphs and multilayer networks; (I.1,I.2,I.3)

A 2.2. Construction of hypergraph models and multilayer network models; (I.2, I.3)

A 2.3. Development of suitable connectivity metrics; (I.2, I.3)

Deliverables:

- research report;
- published articles:
  1. Gaskó, N., Képes, T., Suciu, M., & Lung, R. I. (2021, September). Critical Node Detection for Maximization of Connected Components: An Extremal Optimization Approach. In International Workshop on Soft Computing Models in Industrial and Environmental Applications (pp. 502-511). Springer, Cham.
  2. Suciu, M. A., Gaskó, N., Képes, T., & Lung, R. I. (2021, September). A Simple Genetic Algorithm for the Critical Node Detection Problem. In International Conference on Hybrid Artificial Intelligence Systems (pp. 124-133). Springer, Cham.
  3. Béczi, E., & Gaskó, N. (2021). Approaching the bi-objective critical node detection problem with a smart initialization-based evolutionary algorithm. PeerJ Computer Science, 7, e750
- accepted article:
  1. Gaskó, N. Suciu, M., Lung. R. I. & Képes, T., An evolutionary approach for critical node detection in hypergraphs. A case study of an inflation economic network., The 21th International Conference on Intelligent Systems Design and Applications, accepted.
- web page: www.cs.ubbcluj.ro/˜gaskonomi/GT-NDNetw

I Implementation

The main contributions of the project are presented in what follows.

I.1 Critical node detection problem

I.1.1 Introduction

The study complex of networks gained a huge interest since the introduction of random scale free networks [4]. Since then, several computational problems have arisen in the attempt to understand better the complex networks, such as the community detection [15], influence maximization [10], link prediction [24], etc. A major class of problems concerning complex networks is the node deletion problem [23] and an important subclass of it is the critical node detection. Nodes in the network may have different

Algorithm 1 Standard EO

```
Initialize s at random;
s_best := s; //s_best preserves the best solution found so far
while a termination condition is not met do
  find i_min component in s with the smallest fitness value
  randomly reassign s_{i_min} in s;
  if f(s) > f(s_best) then
    s_best := s
  end if
end while
```

importance according to different network measures. The identification of nodes with highest importance is a challenging computational task depending on the type of network and measure considered.

In general, the critical node detection problem (CDNP) consists in finding a set of k nodes in a given graph $G = (V, E)$, which deleted maximally degrades the graph according to a given measure σ. The CDNP gained popularity because of its large applicability, for example in network immunization [19], network risk management [3], social network analysis [9], etc. There are several studies that focus on the measure σ using different network centrality measures, such as betweenness centrality, closeness centrality, page rank [20, 25].

I.1.2 Related work

In the general formulation of the critical node detection problem we seek to find nodes which are more important than others with respect to a predefined measure. Several graph properties were studied as measures, e.g. the pairwise connectivity which needs to be minimized by deleting k nodes (which is one of the most popular), or minimizing the largest component size by deleting k nodes. In [2] three versions of the CNDP are studied: minimizing the largest components size, minimizing the pairwise connectivity and maximizing the number of connected components.

In this paper we focus on the problem introduced in [32, 31], which is also described in [2]. The problem consists in removing k nodes such that the number of remaining components to be maximal (we denote this problem *kMaxComp*). Formally, if S denotes the set of deleted nodes, and $\mathcal{H}(G[V \setminus S])$ denotes the set of components of graph G without the set of nodes S, the *kMaxComp* problem consists in:

$$\max_{S \subset V} |\mathcal{H}(G[V \setminus S])|,$$
$$\text{such that } |S| \leq k,$$

where $|A|$ denotes the cardinality of set A.

As the CDNP was proved to be NP-hard for several connectivity measures [32] different solving methods has to be proposed. However, *kMaxComp* did not get very much attention. In [32] a Mixed Integer linear programming approach is proposed, and we find a a general integer programming framework in [34]. For a special class of graphs (trees and series-parallel graphs) a dynamic programming approach is designed [31]. In [2] a genetic algorithm is described to solve the problem and two greedy algorithms are presented.

I.1.3 Noisy Extremal Optimization

Extremal optimization (EO) [7, 8] is a simple and powerful combinatorial optimization algorithm which was successfully adapted for different practical problems, e.g. graph partitioning [6], load balancing problem [12]. A variant of EO called NoisyEO [26] was used successfully for the community detection problem.

In the standard variant of the EO two individuals are used during the search: s and s_{best}; s_{best} preserves the best solution found so far by s based on an overall fitness $f()$. EO individuals are represented as composed of several components that can be evaluated separately. The standard EO maximizes each component of a potential solution by randomly replacing the one with the worst fitness. The outline of the standard EO is presented in Algorithm 1.

Algorithm 2 *CN-EO* algorithm

Parameters:

- Probability of shift - p_{shift};
- Number of generations between switching networks - G;
- Total number of generations - $NrGen$;

Randomly initialize (s, s_{best});
repeat
  if s_{best} does not change in G generations and there is no noise **then**
    Induce noise with probability p_{shift};* ;
    Reinitialize s_{best} with the current s value
  end if
  if There has been noise for G generations **then**
    Return to the original network
  end if
  Perform search using the current network (CN-EO(s, s_{best}) iteration);
until Maximum number of generations;
Return s_{best} with highest ***fitness*** achieved on a non-noisy network.

* Modify network by randomly deleting edges with probability p_{shift}

Algorithm 3 CN-EO(s, s_{best}) iteration

For current configuration s evaluate $f_i(s)$, $i \in \{1, \ldots, k\}$.
Find the node with the worst fitness and replace it randmly with another one;
if $(f(s) > f(s_{best}))$ **then**
  set $s_{best} := s$.
end if

In [26] NoisyEO is presented as a variant of EO that proposes the use of a network shifting mechanism to induce diversity in the search. We employ the same mechanism to escape local optima for the CDNP problem. The network shift consists in randomly deleting edges in the network with a probability p_{shift} whenever the search stagnates. The search takes place for a number of generations G on the shifted network, moving the solution away from a local optimum. We call this approach Critical Nodes - EO (CN-EO). The outline of CN-EO is presented in Alg. 2. A step of the CN-EO is outlined in Alg. 3.

Encoding Individual s is represented as a vector of integers of size k representing the critical nodes. Values are from the range 1 and the number of nodes in the graph.

Fitness function Within CN-EO there are two fitness functions used: one to evaluate individual s and a different one for evaluating each component of s. The overall fitness value of $s = (s_1, \ldots, s_k)$ is computed as the number of components of the graph, after removing the k nodes:

$$f(s) = |\mathcal{H}(G[V \setminus \{s_1, \ldots, s_k\}])| \tag{1}$$

The fitness value of a node i in s is computed as its marginal contribution to $f(s)$:

$$f_i(s) = f(s) - f(s \setminus i), \tag{2}$$

where $s \setminus i$ denotes the set of nodes in s without node i.

Network shift The network shift is used to induce diversity in the search by modifying the search space instead of the EO individuals. After a number of generations G the EO search stagnates (there in no change in s_{best}) the network is modified by removing edges with a probability p_{shift}. To preserve some information, the search of s continues, but s_{best} is randomly reinitialized in order to be easily replaced by the new values discovered by s. The search on the modified network takes place for G generations, after which it is resumed on the original network.

I.1.4 Numerical experiments

Parameter settings CN-EO has only three specific parameters: MaxGen is set to 5000, G is set to 10 and p_{shift} to 0.01. The effect of varying these parameters is expected to be similar to that found in [26] and is not a subject of this study.

Benchmarks A set of synthetic benchmarks [1] for the CNDP problem were proposed in [33]. The benchmark set contains three different type of graphs: Barabási-Albert (BA), Erdős-Rényi (ER) and Forest-fire (FF) graphs. BA graphs are scale free networks, ER graphs are random networks, FF graphs simulate how fire spreads through a forest.

Table 1 describes basic network measures of the used benchmarks: number of nodes ($|V|$), number of edges ($|E|$), average degree ($\langle d \rangle$), density of the graph (ρ), and average path length (l_G). Table 2 describes the set of real networks used for numerical experiments, with the same measures as in the case of the synthetic networks. Real networks are from different research areas, e.g biological networks (Ecoli, HumanDis).

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
BA2500	2500	2499	1.999	0.001	6.901
BA5000	5000	4999	2.000	0.000	8.380
ER235	235	350	2.979	0.013	5.338
ER466	466	700	3.004	0.006	5.973
ER941	941	1400	2.976	0.003	6.558
ER2344	2344	3500	2.986	0.001	7.516
FF250	250	514	4.112	0.017	4.816
FF500	500	828	3.312	0.007	6.026
FF1000	1000	1817	3.634	0.004	6.173
FF2000	2000	3413	3.413	0.002	7.587

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	[29]
Circuit	252	399	3.167	0.012	5.806	[27]
EColi	328	456	2.780	0.008	4.834	[35]
USAir97	332	2126	12.807	0.038	2.738	[30]
HumanDis	516	1188	4.605	0.008	6.509	[18]
EUFlights	1191	31610	53.081	0.044	2.622	[28]

Comparison with other methods For comparisons we use three algorithms described in [2]: a greedy algorithm (G_1) based on node deletion from the candidate critical node set, another greedy algorithm (G_2), based on the node addition to the candidate critical node set and a genetic algorithm from an evolutionary algorithm framework using greedy rules (GA). The proposed genetic algorithm uses problem specific variation operators and it is combined with a local search mechanism.

Table 3 presents obtained results for synthetic and real world datasets. Mean value, standard deviation minimum and maximum value are reported over ten independent runs. We also indicate the best known results obtained in [2] and the corresponding method. It can be observed that CN-EO finds the maximum number of connected components when compared to the best known results from the literature. On average CN-EO is better than the best known results for seven datasets.

In Fig. 1 the evolution of the number of connected components in 5000 generations is presented for the BA500 network to illustrate the difference in evolution when using the network shifting mechanism.

Figure 2 depicts the smallest test network, the bovine network, which represents protein interactions, and the obtained critical nodes by CN-EO. It can be seen that CN-EO identifies the nodes that, if removed, generated the maximum number of connected components in the network. We find that in this

case they are also the nodes with the highest degree, but this is not always the case for all networks. We notice that the problem has multiple solutions that lead to the same maximal number of connected components for $k = 3$.

Table 3: CN-EO results. Best known results indicate the best value reported by other methods as well as the method reporting it.

		CN-EO				Best known result
Graph	k	Mean	Std.dev.	Min	Max	(from literature)
BA500	50	**314,8**	1,75	313	318	313 (GA, G_1, G_2)
BA1000	75	**594,5**	5,46	585	604	590 (GA, G_1, G_2)
BA2500	100	1091,3	37,74	1023	1146	**1129** (GA, G_1, G_2)
ER235	50	66,7	1,83	65	71	**68** (GA)
ER466	80	108,9	3,45	102	114	**110** (GA)
ER941	140	194,4	7,73	184	207	**206** (GA)
FF250	50	**93**	1,15	92	95	92 (GA, G_1, G_2)
FF500	110	201,3	36,02	99	217	**215** (GA)
FF1000	150	322,8	13,50	298	344	**340** (GA)
Bovine	3	**77,6**	0,70	77	79	77 (GA, G_1, G_2)
Circuit	25	**31,4**	2,95	28	36	31 (GA)
EUFlights	119	60,5	76,02	11	206	**211** (GA, G_2)
Ecoli	15	**171,8**	2,10	169	175	169 (GA, G_1)
USair97	33	40,6	44,84	3	106	**104** (GA, G_2)
HumanDis	53	**152,7**	3,89	148	158	148 (GA, G_2)

A stock market network application There is a grown interest in constructing and and analyzing economic networks. Nodes can represent banks, directors, investigators, depending on the studied problem [13]. One of the first applications is the stock market analysis from a network perspective [11]. In [17] the Chinese stock market is analysed as a directed network with an influential model. We analyse the financial stock market network described in [22] from the critical nodes perspective. The network is obtained from the analysis of temporal correlations among the time-series of 62 stocks in the New York Exchange Market from the period 2012-2014. We use an unweighted version of the graph where we deleted all edges with weight smaller than 1.2 (edges mean the distance calculated based on the Pearson coefficient). The obtained a network with 62 nodes and 618 edges.

We obtained by NoisyEO the first 3, 4, 5, 6, 7 and 8 most critical nodes. The results are analysed also from a network measure perspective: degree of the nodes, betweenness and closeness centralities were obtained. We indicate the rank of the nodes ordered decreasing. The novelty of searching for critical nodes in a stock market example consists in finding important nodes, other ones as based on the above mentioned traditional measures. As seen in Table 4 based on the critical nodes we can set up another order of importance of the certain stock.

Table 4: Critical nodes and network centrality measures in the New York Exchange Market

Node no.	Stock Name	Critical node (k)	Degree (rank)	Closeness Centrality (rank)	Betweenness centrality (rank)
52	S	k=3	61 (1)	1 (1)	301.62 (1)
51	SO	k=3	48 (3-4)	0.82 (3-4)	164.151 (2)
25	ETR	k=3	48 (3-4)	0.82 (3-4)	140.60 (3)
6	AVP	k=4	51 (2)	0.85 (2)	127.76 (4)
54	UIS	k=5	44 (5-6)	0.78 (5-6)	68.25 (6)
59	WMT	k=6	44 (5-6)	0.78 (5-6)	82.50 (5)
5	ARC	k=7	41 (7)	0.75 (7)	52.61 (7)
1	AEP	k=8	38 (9)	0.72 (9)	52 (8)

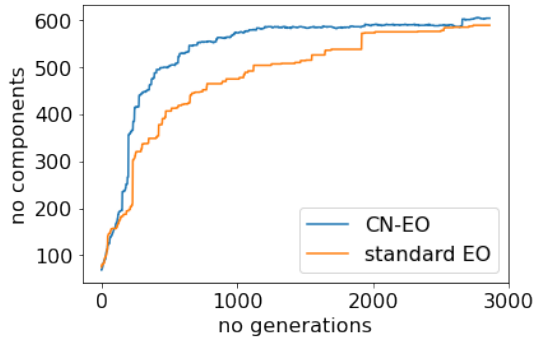


Figure 1: Comparison of evolution the number of connected components obtained by CN-EO and a standard EO for the BA1000 graph in a single run

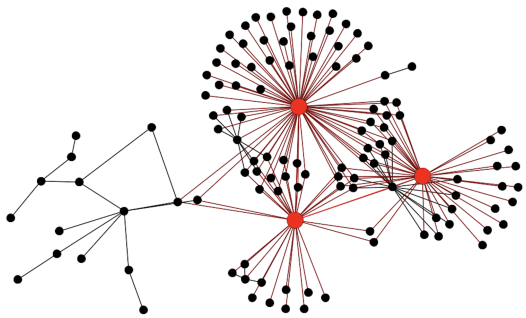


Figure 2: Bovine network (obtained critical nodes are marked with red, $k = 3$)

I.2 Critical node detection in hypergraphs

I.2.1 Introduction

The critical node detection problem (CNDP) [23] is a central topic in graph theory due to its large applicability in various fields, such as immunization [19], network vulnerability [20, 25], economics [3], social network analysis [9], etc. The problem arises from the fact that a node's importance varies in a graph and consists in finding sets of nodes that, when deleted from the graph, maximize a given network measure, which is usually related to the connectivity of the network.

In a general approach, given a graph $G = (V, E)$, the critical node detection problem consists in finding a set S of k nodes which deleted maximally degrades the graph according to a given measure $\sigma(G)$. Examples of such measures include the minimization of pairwise connectivity, minimizing the size of the largest component, bound the pairwise connectivity by a threshold, maximize the number of connected components, etc [21].

While there are many studies that deal with the CNDP for weighted or unweighted, directed or undirected graphs, less attention has been given to the formulation and adaptation of the problem for hypergraphs. A hypergraph generalizes the concept of graph by considering edges that connect more than two nodes. In this setting there are many possible measures that can be considered in the formulation of the CNDP.

In this report we extend the CNDP for hypergraphs by considering the problem of removing k nodes in order to maximize the number of remaining components. Our goal is to use a genetic algorithm adapted to solve this problem. A macroeconomic inflation hypergraph is constructed and analysed in order to illustrate the applicability of the approach.

I.2.2 Critical node detection in hypergraphs

A hypergraph [5] is a generalization of a graph, where edges can join not only two, but any number of nodes. In recent years several application possibilities appeared, where hypergraphs can be used with success, like image classification [36], artificial intelligence [14], biology [16].

Formally, a hypergraph is a $\mathcal{H} = (X, \mathcal{D})$ double, where $X = \{x_1, x_2, ..., x_n\}$ is the set of nodes, $\mathcal{D} = \{D_1, D_2, ..., D_m\}$ is a set of the subsets of X, denoting the set of hyperedges. A simple example

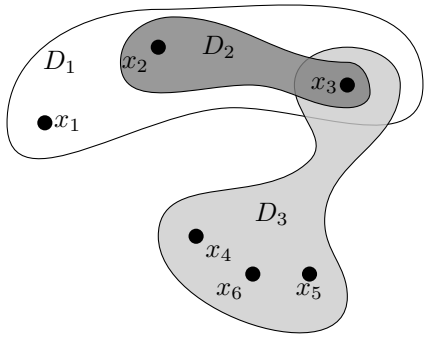


Figure 3: A simple example of hypergraph with six nodes and three hyperedges. If x_3 is deleted, the hypergraph will have three components.

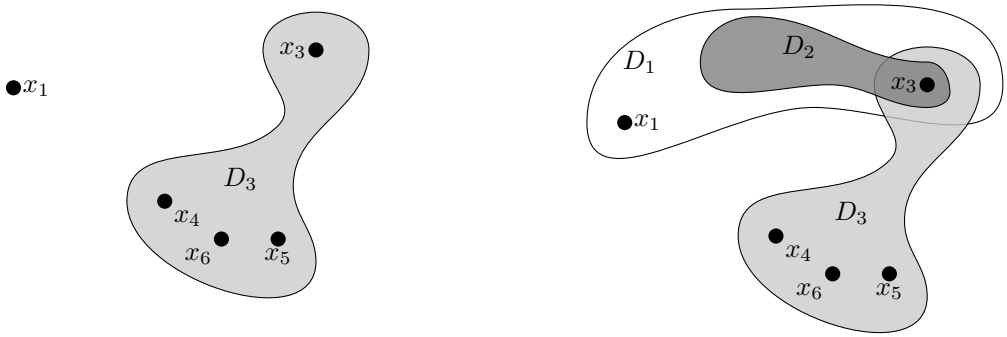


Figure 4: Strong detection of x_2 (left) and weak deletion of x_2 (right) from the hypergraph presented in Fig. 3

is depicted in Figure 3. The hypergraph has six nodes, $X = \{x_1, x_2, x_3, x_4, x_5, x_6\}$, $D = \{D_1, D_2, D_3\}$, $D_1 = \{x_1, x_2, x_3\}$, $D_2 = \{x_2, x_3\}$, $D_3 = \{x_3, x_4, x_5, x_6\}$

In the case of the hypergraphs two variants of node deletion exists: strong and weak deletion. Strong deletion means that not only the node x is removed, but also all edges containing that node. In the case of the weak deletion only the node x is removed. A simple example is illustrated in Figure 4.

The critical node detection problem for the hypergraphs can be formulated as follows: given a $\mathcal{H} = (X, \mathcal{D})$, the problem consists in removing weakly k nodes such that the number of remaining components to be maximal. Formally, if S denotes the set of deleted nodes, and $\mathcal{C}(\mathcal{H}[X \setminus S])$ denotes the set of components of graph $\mathcal{H}$ without the set of nodes S, the problem consists in:

$$\max_{S \subset X} |\mathcal{C}(\mathcal{H}[X \setminus S])|,$$
$$\text{such that } |S| \leq k,$$

where $|A|$ denotes the cardinality of set A.

Example Considering the hypergraph presented in Figure 3, if $k = 1$ x_3 need to be deleted to maximize the number of remaining components.

I.2.3 Hyp-GA

Evolutionary algorithms are powerful optimization tools to solve computationally hard problems. In the next we will use a simple genetic algorithm to solve the critical node detection problem. For the hypergraph we use a clique representation, which means that a hyperedge is transformed to 'traditional' edge (e.g. if a hyperedge has 4 nodes, there will be an edge between each node). Table 5 presents the elements of the genetic algorithm, while specific settings are described in what follows.

Table 5: Hyp-GA Genetic operators

Mutation:	flip-bit mutation
Crossover:	two point crossover
Selection:	tournament selection (tournament size 3)

Algorithm 4 Hyp-GA outline

Initialize population P of size p_{size} at random.
for i=1 to MaxGen **do**
  $P^{(i)}$ = Select p_{size} individuals for variation;
  Offsping= variation operators on $P^{(i)}$;
  Correct and Evaluate Offspring;
  $P^{(i)}$ = offspring;
end for

Encoding An individual x is encoded as a bit string of size equal to the size of the network. Each node is represented in x either as 0 or 1, the value 1 indicates that the corresponding node is included in the set of critical nodes encoded by x. We denote this set by S_x.

Fitness function To evaluate the fitness of an individual we remove all nodes in the critical set it encodes, i.e. nodes with value 1 and count the number of connected components in the remaining graph. We have:

$$f(x) = |\mathcal{H}(G[V \setminus S_x])|. \tag{3}$$

Constraint handling To maintain the size of S_x less that the threshold k we remove excess nodes by using the marginal contribution of a node to the fitness of the individual. The marginal contribution of a node i from the critical set S_x of x, denoted by $u_i(x)$ is:

$$u_i(x) = f(x) - |\mathcal{H}(G[V \setminus \{S_x \setminus \{i\}\}])|,$$

where $f(x)$ is the fitness defined in eq. (3). $u_i(x)$ measures the difference between the fitness of x and the fitness of x when removing node i from S_x. Nodes with lowest marginal contributions are removed from S_x, such that only k nodes remain in the set before each individual evaluation.

I.2.4 Numerical experiments

For numerical experiments we tested the proposed method on synthetic data and on a new macroeconomic dataset.

Synthetic networks

Benchmarks In order to construct hypergraphs with known properties we started from benchmarks constructed for the overlapping community detection problem. LFR benchamrks[1] are baseline test graphs for community detection algorithms. The communities in the graphs show similarity to a hyperedge in a hypergraph, as communities can contain several nodes that are more connected. We generate test graphs of 150 nodes with different characteristics: the proportion of edges that a node has with nodes from the same community $\{0.1, \ldots, 0.6\}$ - low values indicate dense communities (nodes are more connected with nodes from the same community), the higher the number the node has more edges in other communities than in the one he belongs; the number of overlapping communities a node belongs to, 2 for our experiments; how many nodes belong in multiple communities, $\{13, 50, 70\}$ nodes from the 150 nodes of the graph. The generated communities are the input hypergraphs of our algorithm.

[1] https://www.santofortunato.net/resources, last accessed 1/9/2021

Table 6: Critical countries found by the proposed approach.

2010-2012	2013-2015	2015-2017	2017-2019
Afghanistan	Bahamas	Antigua and Barbuda	Burundi
Bangladesh	Guinea	Aruba	Ethiopia
Bhutan	India	Brazil	Haiti
Ghana	Jamaica	Dominica	Iran
India	Japan	France	Jamaica
Iran	Jordan	Greece	Lebanon
Jamaica	Latvia	Iraq	Luxembourg
Moldova	Luxembourg	Israel	Morocco
Morocco	St. Kitts and Nevis	Japan	Sri Lanka
Spain	St. Lucia	Luxembourg	United Kingdom

Parameters For our experiments we use the following parameters for the genetic algorithm are: population size $\{25, 40, 100\}$, maximum number of generations 500, crossover $\{0, 0.5, 0.8\}$ and mutation probability 0.5, probability to mutate a bit $\{0, 0.01, 0.02, 0.03\}$, and tournament size of 3. The values tested for k are 5 and 10.

Results Results are illustrated as boxplots of maximum number of components reported by the algorithm for each parameter setting in figure 5. We find that they are consistently converging to the same number of components across different parameter values, except when using a mutation rate of 0, indicating the known importance of this operator. When more nodes overlap, we find less components for this value of k, as there are more links across communities in the benchmarks. However, results show that the approach is robust, converging in various settings and may be further extended to practical applications.

Inflation hypergraph As a new application we constructed a hypergraph from the world inflation rate (consumer prices). Data about 123 countries is publicly available[2] and contains information about inflation rate from 1960 until 2019. We analyse only the last ten years, $2010 - 2019$, and we eliminate the countries with missing information, after this preprocessing 98 countries remain. The hypergraph is built in the following way: the countries are the nodes of the hypergrah, a hyperedge exists between nodes if they have inflation values in the same year within an interval (one hyperedge contains nodes/countries that have negative inflation for the studied year, the other hyperedges contain the nodes with inflation in the intervals 0-5, 5-10, 10-15 and greater then 15. Because we do not want to obtain one component we divided the 10 years in four intervals (2010-2012,2013-2015,2015-2017,2017-2019) thus obtaining four hypergraphs. The main advantage of this construction is that in this hypergraph the whole dynamics of the inflation of countries appear, because in a single hypergraph several values for each year can be presented. Using the proposed approach we search for the ten most critical nodes in each hypergraph, we present the result in table 6.

A hyperedge in the graph means that countries have similar inflation rate in a certain year. Critical nodes can appear if in a country or in several countries from the same hyperedge the inflation rate changes, thereby it can give a glimpse in a collective set of unstable countries, the countries are investigated as a community.

A new problem, the critical node detection problem for hypergraphs is proposed and a genetic algorithm to solve it. New benchmarks are constructed and as an application a macroeconomic dataset is transformed to a hypergraph, and critical nodes are obtained, which can reveal new information about the dataset.

Experiments demonstrate the potential of the proposed method. As further work other variants of hypergraph critical node detection problem will be studied.

I.3 Critical node detection in multilayer networks

We introduce formally the multilayer and multiplex networks.

Definition A single layer network (a graph) is a tuple $G = (V, E)$, where V is the set of nodes and E is the set of edges $E \subseteq V \times V$.

[2] https://dice.ifo.de/en/node/358439, last accessed 20/09/2021

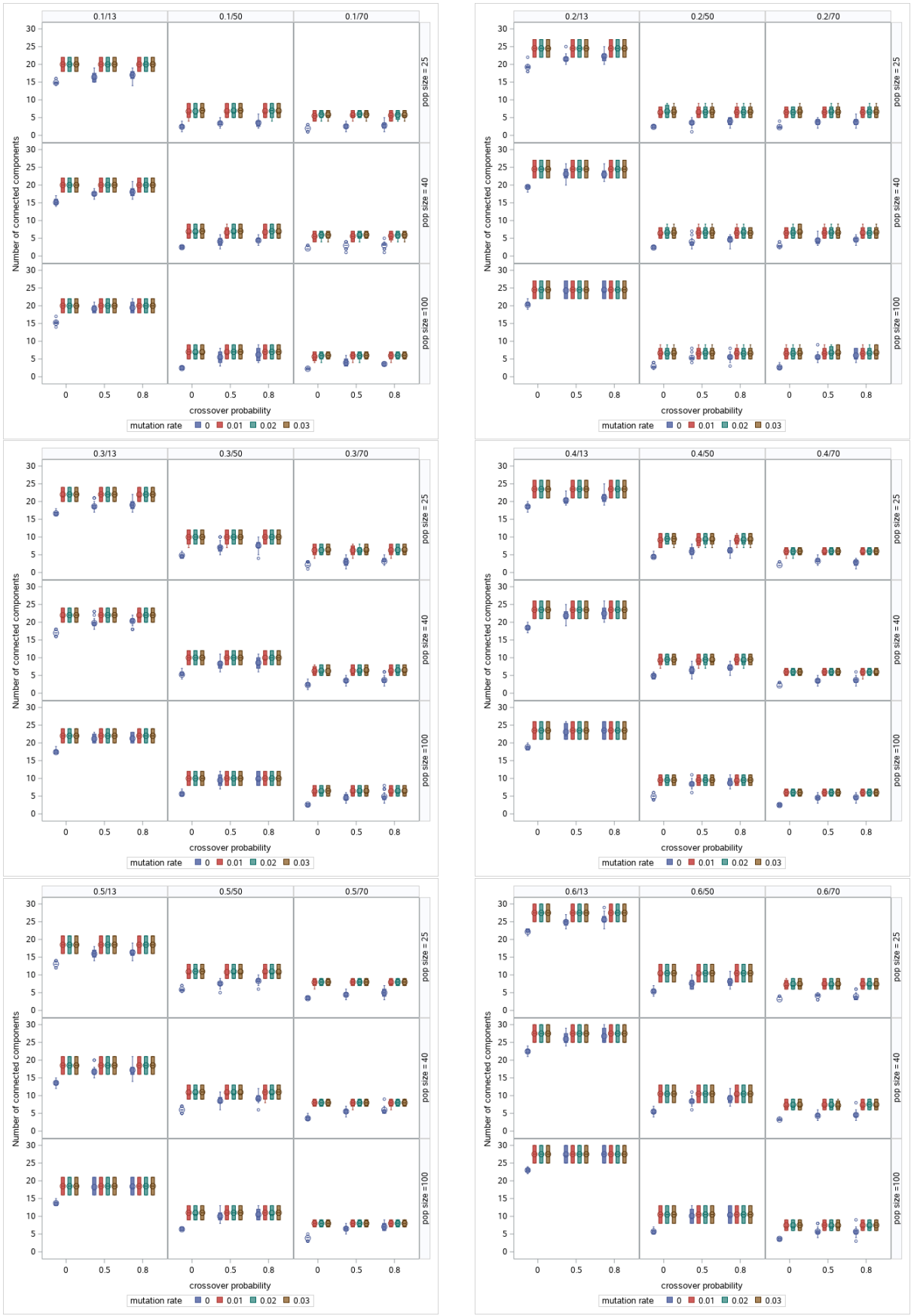
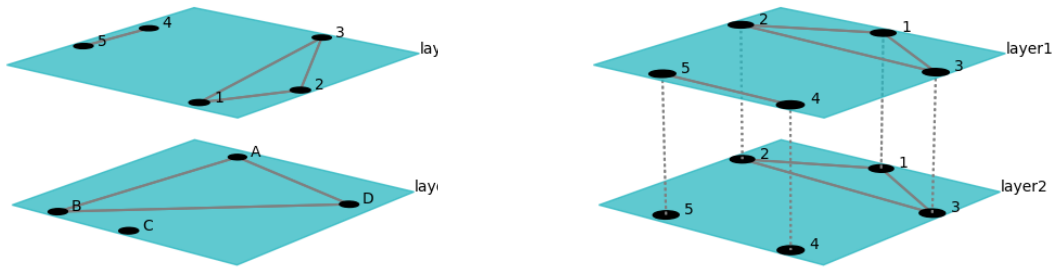


Figure 5: Maximumm number of connected components reported for different hypergraph settings and different parameters.



(a) A multilayer network

(b) A multiplex network

Figure 6: Examples of multilayer and multiplex networks. The multi layer network has a more general structure.

Definition A multilayer network can be defined as a quadruplet $M = (V_M, E_M, V, L)$ [**?**], where V is the set of nodes, $L = \{L_a\}_{a=1}^{d}$ is the set of layers defined by d aspects (if $d = 0$ M reduces to a single layer network, if $d = 1$ M reduces to a multiplex network), $V_M \subseteq V \times L_1 \times \cdots \times L_d$ is the set of node-layer combinations, $E_M \subseteq V_M \times V_M$.

Figure 6 presents a simple example of a multilayer and multiplex network with two layers.

I.3.1 Problem formulation

We introduce two variants of the CND for multilayered networks:

In the first variant the problem consists in removing k nodes such that the number of remaining components to be maximal. Formally, if S denotes the set of deleted nodes, and $\mathcal{H}(M[V \setminus S])$ denotes the set of components of multi layered graph M without the set of nodes S, the problem consists in:

$$\max_{S \subset V} |\mathcal{H}(M[V \setminus S])|,$$

$$\text{such that } |S| \leq k,$$

where $|A|$ denotes the cardinality of set A.

The second variant of the problem consists in removing k nodes such that the size of the largest components to be minimized. Formally, if S denotes the set of deleted nodes, and $\mathcal{H}(M[V \setminus S])$ denotes the set of components of multi layered graph M without the set of nodes S, and M_{max} represents the largest component, the problem consists in:

$$\min_{S \subset V} |M_{max}|,$$

$$\text{such that } |S| \leq k,$$

where $|A|$ denotes the cardinality of set A.

References

[1] www.individual.utoronto.ca/mventresca/cnd.html.

[2] R. Aringhieri, A. Grosso, P. Hosteins, and R. Scatamacchia. A general evolutionary framework for different classes of critical node problems. *Engineering Applications of Artificial Intelligence*, 55:128–145, 2016.

[3] A. Arulselvan, C. W. Commander, P. M. Pardalos, and O. Shylo. Managing network risk via critical node identification. *Risk management in telecommunication networks, Springer*, 2007.

[4] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[5] C. Berge. *Hypergraphs: combinatorics of finite sets*, volume 45. Elsevier, 1984.

[6] S. Boettcher and A. G. Percus. Extremal optimization for graph partitioning. *Physical Review E*, 64(2):026114, 2001.

[7] S. Boettcher and A. G. Percus. Optimization with Extremal Dynamics. *Physical Review Letters*, 86:5211–5214, 2001.

[8] S. Boettcher and A. G. Percus. Extremal optimization: an evolutionary local-search algorithm. In *Computational Modeling and Problem Solving in the Networked World*, pages 61–77. Springer, 2003.

[9] S. P. Borgatti. Identifying sets of key players in a social network. *Computational & Mathematical Organization Theory*, 12(1):21–34, 2006.

[10] W. Chen, Y. Wang, and S. Yang. Efficient influence maximization in social networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 199–208, 2009.

[11] K. T. Chi, J. Liu, and F. C. Lau. A network perspective of the stock market. *Journal of Empirical Finance*, 17(4):659–667, 2010.

[12] I. De Falco, E. Laskowski, R. Olejnik, U. Scafuri, E. Tarantino, and M. Tudruj. Extremal optimization applied to load balancing in execution of distributed programs. *Applied Soft Computing*, 30:501–513, 2015.

[13] F. Emmert-Streib, S. Tripathi, O. Yli-Harja, and M. Dehmer. Understanding the world economy in terms of networks: A survey of data-based network science approaches on economic networks. *Frontiers in Applied Mathematics and Statistics*, 4:37, 2018.

[14] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao. Hypergraph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3558–3565, 2019.

[15] S. Fortunato. Community detection in graphs. *Physics reports*, 486(3-5):75–174, 2010.

[16] N. Franzese, A. Groce, T. Murali, and A. Ritz. Hypergraph-based connectivity measures for signaling pathway topologies. *PLoS computational biology*, 15(10):e1007384, 2019.

[17] Y.-C. Gao, Y. Zeng, and S.-M. Cai. Influence network in the chinese stock market. *Journal of Statistical Mechanics: Theory and Experiment*, 2015(3):P03017, 2015.

[18] K.-I. Goh, M. E. Cusick, D. Valle, B. Childs, M. Vidal, and A.-L. Barabási. The human disease network. *Proceedings of the National Academy of Sciences*, 104(21):8685–8690, 2007.

[19] J. He, H. Liang, and H. Yuan. Controlling infection by blocking nodes and links simultaneously. In *International workshop on internet and network economics*, pages 206–217. Springer, 2011.

[20] S. Iyer, T. Killingback, B. Sundaram, and Z. Wang. Attack robustness and centrality of complex networks. *PloS one*, 8(4):e59613, 2013.

[21] M. Lalou, M. A. Tahraoui, and H. Kheddouci. The critical node detection problem in networks: A survey. *Computer Science Review*, 28:92–117, 2018.

[22] V. Latora, V. Nicosia, and G. Russo. *Complex networks: principles, methods and applications.* Cambridge University Press, 2017.

[23] J. M. Lewis and M. Yannakakis. The node-deletion problem for hereditary properties is np-complete. *Journal of Computer and System Sciences*, 20(2):219 – 230, 1980.

[24] D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031, 2007.

[25] M. Lozano, C. García-Martínez, F. J. Rodriguez, and H. M. Trujillo. Optimizing network attacks by artificial bee colony. *Information Sciences*, 377:30–50, 2017.

[26] R. I. Lung, M. Suciu, and N. Gaskó. Noisy extremal optimization. *Soft Computing*, 21(5):1253–1270, 2017.

[27] R. Milo, S. Itzkovitz, N. Kashtan, R. Levitt, S. Shen-Orr, I. Ayzenshtat, M. Sheffer, and U. Alon. Superfamilies of evolved and designed networks. *Science*, 303(5663):1538–1542, 2004.

[28] T. Opsahl. Why anchorage is not (that) important: Binary ties and sample selection.

[29] J. Reimand, L. Tooming, H. Peterson, P. Adler, and J. Vilo. Graphweb: mining heterogeneous biological networks for gene modules with functional significance. *Nucleic Acids Research*, 36:452–459, 05 2008.

[30] R. A. Rossi and N. K. Ahmed. The network data repository with interactive graph analytics and visualization. In *AAAI*, 2015.

[31] S. Shen and J. C. Smith. Polynomial-time algorithms for solving a class of critical node problems on trees and series-parallel graphs. *Networks*, 60(2):103–119, 2012.

[32] S. Shen, J. C. Smith, and R. Goli. Exact interdiction models and algorithms for disconnecting networks via node deletions. *Discrete Optimization*, 9(3):172–188, 2012.

[33] M. Ventresca. Global search algorithms using a combinatorial unranking-based problem representation for the critical node detection problem. *Computers & Operations Research*, 39(11):2763–2775, 2012.

[34] A. Veremyev, O. A. Prokopyev, and E. L. Pasiliao. An integer programming framework for critical elements detection in graphs. *Journal of Combinatorial Optimization*, 28(1):233–273, 2014.

[35] R. Yang, L. Huang, and Y.-C. Lai. Selectivity-based spreading dynamics on complex networks. *Physical review e*, 78(2):026111, 2008.

[36] J. Yu, D. Tao, and M. Wang. Adaptive hypergraph learning and its application in image classification. *IEEE Transactions on Image Processing*, 21(7):3262–3272, 2012.

Project director,
Conf. dr. Gaskó Noémi