ISSN: 1803-3814 (Printed), 2571-3701 (Online)
https://doi.org/10.13164/mendel.2019.2.011

HOW TO BURN A NETWORK OR SPREAD ALARM

Marek Šimon[✉], **Ladislav Huraj, Iveta Dirgová Luptáková, and Jiří Pospíchal**

Department of Applied Informatics, Department of Applied Informatics, Slovak Republic
marek.simon@ucm.sk[✉], ladislav.huraj@ucm.sk, iveta.dirgova@ucm.sk, jiri.pospichal@ucm.sk

Abstract

This paper compares centrality indices usage within a heuristic method for a fast spread of alarm, or crucial information. Such indices can be used as a core part within more sophisticated optimisation methods, which should determine a graph parameter - burning number, defining, how fast can an alarm spread through all nodes. In this procedure at each time step a new chosen node is alarmed (i.e. burned) "from outside", and already alarmed nodes at each time step then alarm their neighbours. The procedure ends, when all the nodes are alarmed (i.e. burned). The optimisation heuristic should choose such ordered sequence of nodes, which are to be alarmed "from outside", that their number, equal the number of time steps (i.e. burning number) necessary to alarm the whole network, is minimised. The NP completeness of the problem necessitates a usage of heuristics. However, even the heuristics can be slower, reaching towards a global optimum, or faster, exchanging part of the quality for a time. This paper studies the usage of centrality indices in a simpler and faster heuristic. It should be useful e.g. for a mobile network of cars or drones, when an optimal solution cannot be computed in advance, or take too much CPU time, since the connections within the dynamic network might not exist any longer. A wide range of centrality indices was tested on selected networks, both real as well as artificially generated. While the performances of indices substantially differ for different types of networks, results show, which centrality indices work well across all tested networks.

Keywords: *centrality indices, burning number, complex networks, optimisation heuristics.*

Received: 02 November 2019
Accepted: 06 December 2019
Published: 21 December 2019

1 Introduction

Spreading an information, be it alarm, traffic management instructions to swarms of air drones or wi-fi connected cars via satellite, or instructions to a land-based group of connected monitoring stations, should be done quickly and effectively. Often, delivery of time-critical information can reduce operational risk [8]. Ideally, the information is sent to the selected nodes (drone, vehicle, station), which then proceed to inform their neighbours, and their neighbours inform their neighbours, while the satellite continues to connect with further nodes, which did not yet obtain the information. This information delivery is finished when all the nodes obtained the information, or, if the information should be renewed periodically, another round of information delivery is started. Whether the information delivery is one-time job or repeating one, it should be done as fast as possible.

Let us, for the sake of simplification, assume that the time it takes to contact one node via satellite (e.g. secure laser connection) is the same as the time necessary for the node to contact all its direct neighbours. The time optimisation task is then reduced to the optimal selection of a minimal sequence of nodes, which are, one at each time step, informed by a satellite. The selection of nodes and their order depends heavily on the size and structure of the network, where two nodes are connected, when they can reach each other through local connection, e.g. wi-fi. Such a problem was already extensively studied from a purely mathematical point of view in graph theory as a burning number problem [5,6,13]. The burning number is the number of time steps (as well as the number of nodes in the optimum sequence), after which all the nodes obtained their information.

A host of similar problems also exists, like k centre selection [9], gossiping [11], broadcasting of control or emergency packets using clustering [2, 12], influence maximisation [16], or firefighter problem [7]. However, their solution cannot be used directly in the burning number problem.

Soon after the burning number problem was formally defined, its solution was proved NP-complete [3]. The above-mentioned graph theorists provided algorithms only to determine the upper and lower boundaries of the solution. While the determination of upper boundaries provided also a solution to the problem in a form of sequence of nodes, it was typically rather suboptimal, as shown in further study aimed at heuristic optimisation of burning number [18]. The heuristics studied in [18] used as its part only one way how to determine a current "center" of the network, based on eigenvector. Moreover, even though the more sophisticated heuristics designed in [18] obtained better solutions by a rather complicated guided search through the search space, they were rather cumbersome and more CPU demanding.

The computational demands of the existing heuristics were the main reason, why in this paper we are trying to use a range of centrality measures apart from eigenvector values. Here, these centrality indices are combined with the simplest and least demanding heuristic approach from [18], in the hope to obtain "quick and dirty" solution, that can be further speeded up by a parallelization [17]. Such fast heuristic could be used e.g. in the mentioned dynamic network of mobile nodes with time-dependent structure. A perfect, but more CPU demanding heuristic might provide a solution, which might be no longer valid, because the network topology changed in the meantime. The selection of the best centrality index might be also used further in more sophisticated approaches instead of the eigenvalue index.

2 Centrality Indices and Burning Number

A networks for the present approach shall be simplified into a graph $G = (V, E)$, which vertices from the vertex set V do not have positions in the Euclidean space and its edges from a set E are not directed or weighted and the graph does not contain loops or multiple edges. The distance $dist_G(u; v)$ of vertices $u, v \in V$ equals the number of edges in a shortest path between them. Formally [6], the burning number $b(G)$ is the minimum integer value k such, that there exists a sequence of vertices $x_1, \dots, x_k$ and any vertex $u \in V$ is within a distance $k - i$ for some vertex x_i from the sequence.

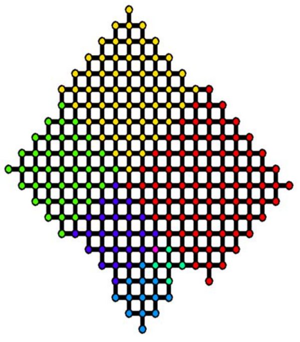

Figure 1: An example of a graph with the burning number 7 (the graph is further referred to as *squaredIdealBurn7*), where the sequence of vertices $x_1, \dots, x_7$ can be specified as the central vertices of the red, yellow, green, dark blue, light blue, light green and maroon sets of vertices, from which the "burning" consecutively spread.

In the Figure 1 are shown by coloured regions the vertices, which obtained their information, directly or through intermediate vertices, from the central vertices of the coloured regions. The centre of the greatest red region was the first in the sequence, so in the second time step it sent its information to its 4 neighbours in the distance 1, while the yellow centre region got its information. In the third step, the 4 red neighbours sent the information to their 8 neighbours, while the yellow centre node sent the information to its 4 neighbours and the centre of the green region got its information. In a similar way the information spreads in 7 steps through the whole network. In the Figure 1 there are no cases, when an uninformed vertex could get an information from 2 "differently coloured" informed neighbours, but it may happen in other cases. However, the colours are shown here only for our information, basically the vertices are divided only into two groups, informed or uninformed, and it does not matter, from which neighbour a vertex got its information (further in indecisive cases the matter is solved randomly).

As can be seen from the Figure 1, it would be advantageous to find out a vertex x_1 somewhere near the centre of the graph, and for a presumed burning number k, the algorithm should remove the vertex together with its neighbourhood up to the distance $k - 1$. Then a second centre, the second entry x_2 for the sequence of $x_1, \dots, x_k$ should be selected and its $k - 2$ neighbourhood removed, etc. The problem is, how to define the centre. In [18] an eigenvector index was used, where a vertex with its maximum value was selected as a next centre for the vertex sequence $x_1, \dots, x_k$. The simple, greedy like version of the algorithm described above did not work ideally. It found out a burning number 10 instead of the ideal burning number 7 illustrated above in the Figure 7. The Figure 7 shows also the reason of this failure. The ideal centres, whatever their definition, are often not wholly ideal, and a slightly "off center" selection partially "tugged" in a corner might be better. Rather complicated algorithm, more extensively searching through the search space, was given in [18].

Therefore, it might be advantageous to try other centrality indices in simpler approaches to find out, if they would not provide better results than eigenvector centrality.

In all the approaches to find the burning number [6, 18], the idea is to try a concrete burning number n, corresponding to a selection of a sequence vertices $x_1, \dots, x_n$, serving as the centres. If the graph is not burned, then try another burning number n with a different sequence of vertices. This search for a proper n is most efficiently done by a binary search, where the goal is to pinpoint a critical number n, for which vertex sequence "burns" or informs the whole graph, but for $n-1$ no such sequence can be found. The difference between a burning number found by a perfect algorithm, and a suboptimum higher estimate of burning number found by a less than perfect algorithm is typically in one or at most several units.

In order to distinguish more distinctly between centrality indices used to find a burning number, in the following description we shall use the "ideal" burning number found out by a lengthy procedure as an estimate, and see, how close the fast imperfect algorithm will come to informing all vertices by using different centrality indexes.
Further, we are not trying to find out a burning number, but to see, how many vertices were left uninformed or "unburned". This number is much more discriminating, since the number of vertices is typically greater by two orders of magnitude, and the number of remaining unburned vertices is usually greater by an order of magnitude than the burning number.

3 The Algorithm Testing Centralities to Minimize the Number of Unburned Vertices

As already stated in the previous section, a typical algorithm aimed to find a burning number consists of two parts: A binary like search for a burning number, where the answers: "The tested number resulted in all vertices "burned"" or "The tested number resulted in some vertices left "unburned"". The second part of the algorithm consists of a function, which tries, for a given number n, to generate such a sequence of vertices $x_1, \dots, x_n$, which should burn as much vertices as possible. The further described algorithm, used for testing, does not give a binary result that the tried number is/is not a burning number, but provides us for a given tested number n with the resulting number of unburned vertices. Of course, if the result would be zero unburned vertices, the tested number n might be the burning number. However, for our comparison, we are more interested with the number of remaining unburned vertices. The testing algorithm, further given in pseudocode as Algorithm 1 is adapted from Algorithm 1 in [18].

Algorithm 1: Heuristic testing centrality to minimize the number of unburned vertices.

```
Input: A network $G = (V, E)$, a guess value $bg$ of a burning number, $centrality$ method
Output: Number of unburned vertices or a sequence $X = x_1, \dots, x_{bg}$ of nodes from $G$
1:  $X \leftarrow \emptyset$ ; $TG \leftarrow G$;
2:  for $i = 1, \dots, bg$ do
3:      $maxcomp \leftarrow$ component of $TG$ with maximum no. of vertices
4:      if $i \geq$ radius of $maxcomp$
5:      then $x_i \leftarrow \mathrm{v} \in \mathrm{V} \mid \min_{v \in V(maxComp) \setminus X} eccentricity(v, maxComp)$
6:      else $x_i \leftarrow \mathrm{v} \in \mathrm{V} \mid extreme_{v \in V(TG) \setminus X}\ centrality(v, maxComp)$
7:      $X \leftarrow \mathrm{X} \cup \mathrm{x_i}$
8:      $V_e(x_i) \leftarrow \mathrm{ego}(\mathrm{G}, \mathrm{bg} - \mathrm{i}, \mathrm{x_i})$
9:      $\mathrm{S_i} \leftarrow \cup_{\mathrm{j=1}}^{\mathrm{j=i}} V_e(x_j)$
10:     $\mathrm{TG} \leftarrow \mathrm{G}(\mathrm{V}, \mathrm{E}(\mathrm{TG}) \setminus \mathrm{E}(\mathrm{G}[\mathrm{S_i}]))$
11: if $S_i \equiv V$ then return $X$ else return $|V \setminus S_i|$
```

The greedy Algorithm 1 contains some notions like component, radius, or eccentricity, that are standard in graph theory, and therefore shall not be explained further. However, the notion extreme centrality requires a more detailed description. Some approaches in centrality measures or indices, like eccentricity or eigenvalues, achieve their maximum value for the vertices in the "centre of the graph", which is the present goal. Other indices, like a famous Wiener index, defined for a current vertex as the sum of the lengths of the shortest paths between the current vertex and all the remaining vertices, have their minimum in the "centre of the graph". The decision, if to use maximum or minimum for a concrete centrality method is therefore not straightforward.

Since the eigenvalue centrality worked in [18] quite well, in further computations for a given graph firstly a correlation between the eigenvalue centrality and the tested centrality was calculated for the whole set of vertices. If the correlation was positive, the word extreme in the line 6 of the Algorithm 1 above was replaced by the function *maximum.* If the correlation was negative between the tested centrality and eigenvalue centrality, the *extreme* in the line 6 was replaced by *minimum.* The function $ego(G, bg - i, x_i)$ provides for a graph G and a graph distance $bg - i$ all the vertices of G, which are within the graph distance $bg - i$ of the vertex x_i.

4 The Results of Tested Centrality Indices Against Tested Networks

4.1 Tested Networks

The ultimate goal of the presented selection of centrality indices is rather practical, that is, the fast spread of information in a mobile network spread on a two-dimensional plane. It is reflected also on the choice of tested networks. Only three networks were selected for testing. The first one is called *squaredIdealBurn7* and it is shown in Figure 1. It is an artificial example purposefully made to be difficult, originated in [18], but it approaches a regular two-dimensional grid network, only with its perimeter slightly irregular. This irregularity makes the placement similar to the problems encountered in reality, where the boundaries of a network are often irregular through the geographical features like presence of mountains or lakes.

The second example, netscience, comes from a repository [15]. It serves as an example of a real (social) citation network, but is not really restricted much by two dimensional positions.

The third tested network is a geometric random network, which was generated artificially, but this type of network is used regularly to simulate the landlocked hubs connections. It starts as a randomly generated 1000 points (vertices) on a square of dimensions 1x1, where two vertices are connected, if their Euclidean distance is smaller than or equal to a given boundary, in our case 0.05. The networks characteristics given in Table 1 substantially differ e.g. in maximum degree or number of triangles reflected by an average clustering coefficient. The networks are therefore expected to produce different test results. The best-found burning number was provided by the good, but complicated, time consuming algorithm from [18]. This was further used as a guess value *bg* of burning number for the above described Algorithm 1.

Table 1: Characteristics of the tested networks

Name of network	Description	$\|V\|$	$\|E\|$	max. degree	min. degr.	avg. degree	density	average clust. coeff.	best found burning
squaredIdealBurn7	artificial example	231	418	4	1	3.62	0.016	0.00	7
netscience	co-authorship	379	914	34	1	4.82	0.010	0.06	6
Geometric random	generated net	1000	3770	19	1	7.54	0.008	0.61	11

4.2 Tested Centrality Indices

In the recent decades, a large number of centrality indices was designed, with their purpose ranging from quantitative structure activity relationships studied for chemical graphs by topological indices, applied analysis of complex networks [14,10] like brain/network analysis, targeted deconstruction of networks by deleting central vertices, finding unofficial influencers to spread political messages, preferential immunisation of people most likely to spread further a disease, up to finding key nodes in the internet.

Methods resulting in functions to calculate such indices were gathered in many software libraries. This paper uses a more recent one, the R language library Central Informative Nodes in Network Analysis aka CINNA [1]. From these there were weeded out numerically unstable functions as well as those, which took too much CPU time. Thus, finally 31 centrality indices were selected. An example of their application is given in Figure 2, where the network *squaredIdealBurn7* served for testing the Algorithm 1 with the guessed burning number 7, using the centrality index subgraph centrality. In the Figure 2 there is presented one of the better results of the subgraph centrality index, leaving us only with 17 unburned vertices, while the previously used eigenvector index resulted in 49 unburned vertices.

Table 2: Rank of achieved average numbers of unburned vertices as well as CPU time on PC for the tested centrality indices

	Method	*SquaredIdealBurn7*	NetScience	Geom. Rand. Net.	CPU Time in sec.
1	Barycenter Centrality	5	7	10	0.16
2	Closeness (Freeman)	6	10	11	0.10
3	Lin Centrality	3	17	3	0.17
4	Dangalchev Close.	8.5	11	2	0.28
5	Residual Closeness	8.5	8	5	0.28
6	Harmonic Centrality	17.5	4	8	0.14
7	Betweenness	24	9	**1**	0.14
8	Decay Centrality	8.5	**1**	24	0.21
9	Average Distance	20	6	6	0.53
10	Wiener Index	22	5	9	0.23
11	Radiality Centrality	4	16	12	0.78
12	Geodesic K-Path	2	13	23	0.41
13	Markov Centrality	8.5	12	7	6.03
14	Degree Centrality	19	23	15	0.03
15	Closeness (Latora)	17.5	20	14	0.16
16	Kleinberg's authority	14	26	19	0.03
17	Cross-Clique Conn.	21	14	4	6.01
18	Diffusion Degree	12	21	17	0.57
19	subgraph centrality	**1**	25	22	1.95
20	Lobby Index	16	15	21	0.60
21	Kleinberg's hub cent.	14	29	29	0.03
22	Eccentricity	28	19	18	0.19
23	eigenvector	14	28	30	0.03
24	Laplacian Centr.	11	22	20	4.28
25	K-core Decomposition	25	27	26	0.03
26	Max. Neigh. Comp.	26	24	13	3.30
27	Leverage Centr.	30	3	28	5.42
28	Group Centrality	29	2	31	4.43
29	Local Bridging Centr.	31	18	16	3.20
30	DMNC Density	23	31	25	3.10
31	Entropy Centr.	27	30	27	11.44

Here it should be mentioned, that in the line 6 of the Algorithm 1 above, where the next vertex is to be selected, one sometimes has to choose from several vertices with the same index value. In that case the selection is made randomly. Since this decision substantially influences the result of the Algorithm 1, in case that such selection occurs, the method was repeated 100 times. This is reflected by the box-and whiskers plot describing the results in Figures 3-5, where in such cases a box and whisker extreme values show us quartiles, and the band inside the box is median, with possible outliers as open circles. However, for some of the indices, like for the eigenvector centrality, the values of vertices are all different in a non-symmetrical graph, resulting in just one value instead of a box and whiskers representation.

The Figure 6 shows the ranks of the number of unburned vertices. Since for *squaredIdealBurn7* network the centrality index, which achieved, on average, the lowest number of unburned vertices, was the subgraph centrality, its rank value shown in red is 1. The methods are ordered by the sum of the achieved ranks, and unfortunately, the results of subgraph centrality for the other two networks are not good, the index is placed in the Figure 6 in the second half. The same order of the centrality methods is in the Table 2, where the best ranking methods are emphasized by red bold numbers. From the Table 2, as well as from the Figure 6 it is apparent, that the best method for one type of network give substandard results for other types of networks. The "best all-purpose" indices like Barycenter or Closeness centrality do not provide real top results for any of the networks. Nevertheless, it is apparent, that some of the methods, like DMNC density or Entropy centrality end up near the "end of the race" for all the networks, as well as in CPU time.

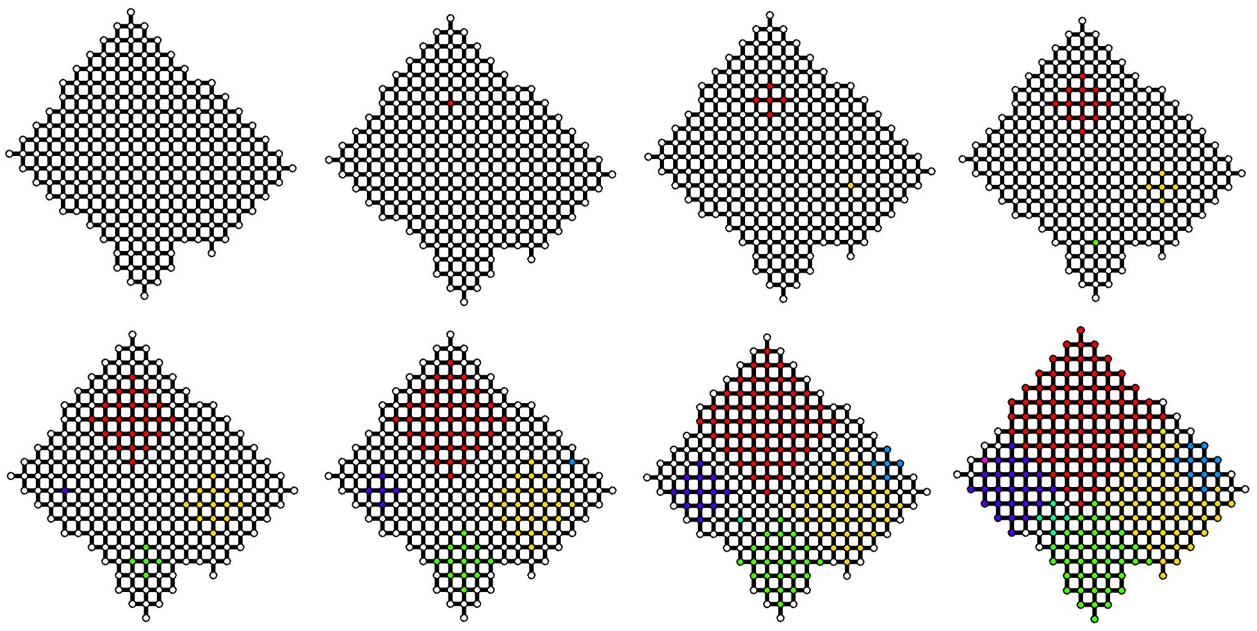

Figure 2: An example of single time steps in the process of the burning of a graph by the Algorithm 1, using the subgraph centrality index on the *squaredIdealBurn7* network already shown in Figure 1. Here, after 7 time steps of the cycle in the line 2 of the Algorithm 1, 17 vertices along the borders of the network marked by a white color were still left unburned.

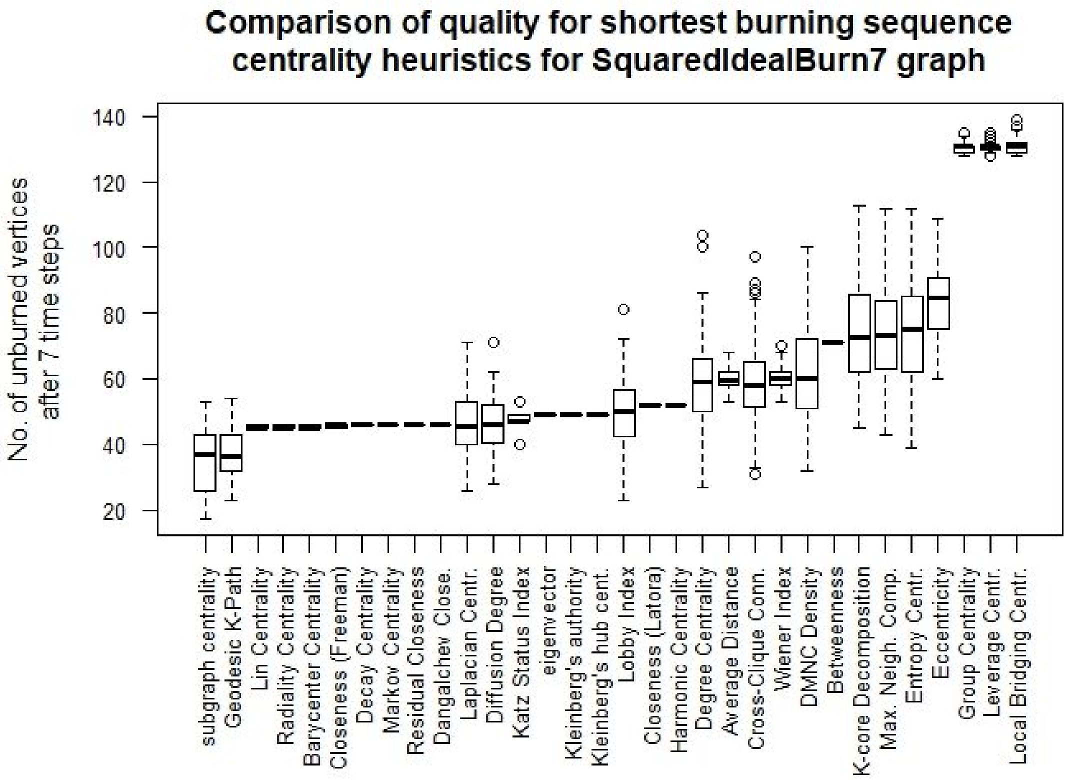

Figure 3: Number of unburned vertices after 7 time steps for *squaredIdealBurn7* network.

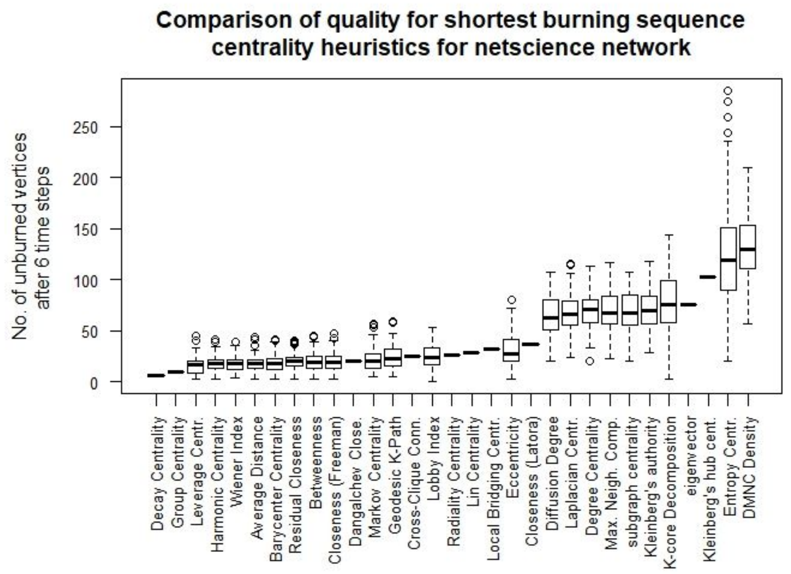

Figure 4: Number of unburned vertices after 67 time steps for netscience network.

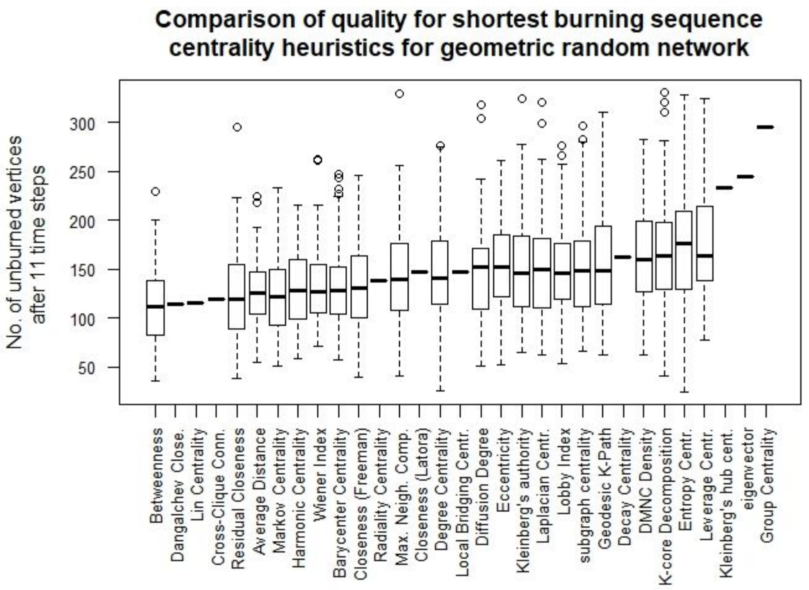

Figure 5: Number of unburned vertices after 11 time steps for a geometric random network.

5 Conclusion

The best all-purpose centrality indices for burning number algorithms seem to be Barycenter and Closeness (Freeman) centrality indices. However, for practical mobile networks, the best model is a geometric random network and for this purpose a well-know betweenness centrality index provides the best results. In a real application, one can assume, that the rough number of vertices - mobile stations is known in advance, from which should roughly follow also the burning number guess. Unlike in the optimisation algorithm, where the algorithm has to start anew, if some vertices are left unburned, in practice after the guessed number of time steps the information continues to spread, so it takes perhaps just a couple of steps more for the information to permeate throughout the whole network. Therefore, even imperfectly selected sequence of centres to be informed “from outside” has a practical value.

Acknowledgement: The work was in part supported by the grant VEGA 1/0145/18 Optimization of network security by computational intelligence, and the grant APVV-17-0116 - Algorithm of collective intelligence.

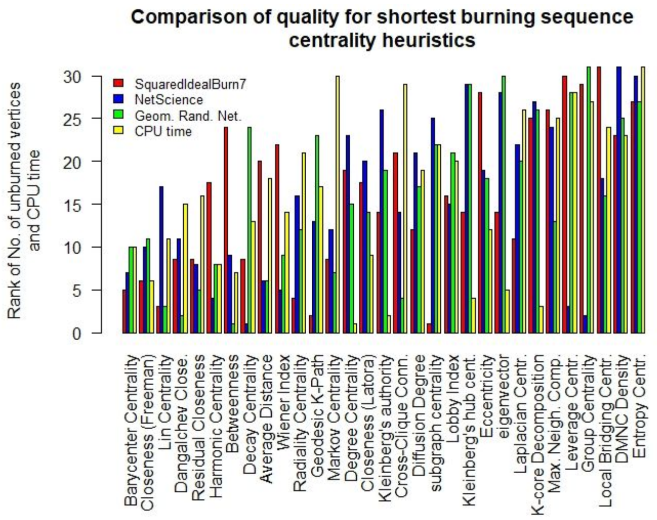

Figure 6: Comprehensive results for all 3 networks, showing rank of the centrality indices for single networks as well as for the CPU time.

References

[1] Ashtiani, M., Mirzaie, M., and Jafari, M. 2018. CINNA: an R/CRAN package to decipher Central Informative Nodes in Network Analysis. *Bioinformatics* 35(8), pp. 1436–1437.

[2] Benkerdagh, S. and Duvallet, C. 2019. Cluster-based emergency message dissemination strategy for VANET using V2V communication. *International Journal of Communication Systems*, 32(5), p.e3897.

[3] Bessy, S., Bonato, A., Janssen, J., Rautenbach, D., and Roshanbin, E. 2017. Burning a graph is hard. *Discrete Applied Mathematics* 232, pp. 73–87.

[4] Bessy, S., Bonato, A., Janssen, J., Rautenbach, D., and Roshanbin, E. 2018. Bounds on the burning number. *Discrete Applied Mathematics* 235, pp. 16–22.

[5] Bonato, A., Janssen, J., and Roshanbin, E. 2016. How to burn a graph. *Internet Mathematics* 12(1-2), pp. 85–100.

[6] Bonato, A. and Kamali, S. 2019. Approximation Algorithms for Graph Burning. In T.V. Gopal, J. Watada (eds.) *Theory and Applications of Models of Computation*, pp. 74–92. Springer, Cham.

[7] Finbow, S. and MacGillivray, G. 2009. The Firefighter Problem: a survey of results, directions and questions. *Australasian J. Combinatorics* 43, pp. 57–78.

[8] Gabriska, D. and Olvecky, M. 2018. Analysis and Risk Reduction in Operation of Hazardous Programmable Electronic Systems. *2018 IEEE 16th International Symposium on Intelligent Systems and Informatics* (SISY), IEEE. DOI: 10.1109/SISY.2018.8524642

[9] Hochbaum, D. S. 1997. Approximation Algorithms for NP-Hard problems. PWS Publishing Company, Boston, pp. 346–398. DOI: 10.1145/261342.571216

[10] Hosťovecký, M. and Babušiak, B. 2017. Brain activity: beta wave analysis of 2D and 3D serious games using EEG. *Journal of Applied Mathematics, Statistics and Informatics* 13, pp. 39–53.

[11] Hromkovič, J., Klasing, R., Monien, B., and Peine, R. 1996. Dissemination of information in interconnection networks (broadcasting & gossiping). In *Combinatorial network theory* (pp. 125–212). Springer, Boston, MA.

[12] Kotyrba, M., Volná, E., and Kominkova-Oplatkova, Z. 2014. Comparison of Modern Clustering Algorithms For Two-Dimensional Data. In *Proceedings 28th European Conference on Modelling and Simulation, ECMS 2014*, pp. 346–351. Brescia, Italy.

[13] Mitsche, D., Prałat, P. and Roshanbin, E. 2017. Burning graphs: a probabilistic perspective. *Graphs and Combinatorics* 33(2), pp. 449–471.

[14] Newman, M. 2018. Networks. Oxford University Press. ISBN: 9780198805090.

[15] Rossi, R. and Ahmed, N. 2015. The network data repository with interactive graph analytics and visualization. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, pp. 4292–4293.

[16] Samadi, M., Nagi, R., Semenov, A., and Nikolaev, A. 2018. Seed activation scheduling for influence maximization in social networks. *Omega* 77, pp. 96–114.

[17] Siládi, V., Povinský, M. and Satymbekov, M. 2017. Adapted parallel Quine-McCluskey algorithm using GPGPU. In *14th International Scientific Conference on Informatics*, IEEE, pp. 327–331.

[18] Šimon, M., Huraj, L., Dirgová-Luptáková, I., Pospíchal, J. 2019. Heuristics for Spreading Alarm throughout a Network. *Appl. Sci.* 9, 3269. DOI: 10.3390/app9163269