



UNIVERSITÀ DEGLI STUDI
DI GENOVA

Department of Computer Science
GA - Graph Analytics A.Y. 2018/2019

Network Robustness

Andrea Canepa (S 4185248)



*Facebook is quite entrenched and has a network effect.
It's hard to break into a network once it's formed.*

Elon Musk

23 April 2020

Contents

1	Introduction	2
2	Erdős-Renyi Random Graph	3
2.1	Random Attack	4
2.2	Closeness Oriented Attack	5
2.3	Betweenness Oriented Attack	6
2.4	Hits Oriented Attack	7
2.5	Clustering Oriented Attack	8
2.6	PageRank Oriented Attack	9
2.7	Summary	10
3	American TV Shows Facebook Pages Network	11
3.1	Random Attack	11
3.2	Closeness Oriented Attack	12
3.3	Betweenness Oriented Attack	13
3.4	Hits Oriented Attack	14
3.5	Clustering Oriented Attack	15
3.6	PageRank Oriented Attack	16
3.7	Summary	17

1 Introduction

During this experiment I tried to mine the **robustness** of two different graphs:

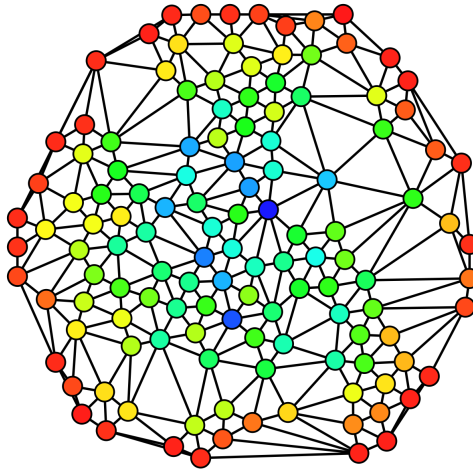
- **small**: an *Erdős-Renyi* random graph, with 500 nodes and a probability of connectivity big enough ($p = 0.02$) to satisfy the **connected regime** hypothesis;
- **big**: the network of *American Tv Shows' Facebook pages*, which is a **social network**.

I performed six different attacks, each one based on a different *metric*, alongside one that affects nodes in a *random* way. To track the *evolution* of the network I decided to consider four different characteristics of the graph such as:

- **size** (absolute and relative) of the giant component
- **diameter**
- **average degree**, i.e. $\langle k \rangle$
- **average** shortest path length

In the deliver, You can find some videos to visualize how the network change after each attack in the directory `imgs/attack/videos/mp4`, built with `ffmpeg`. There is also a *Bash* utility, named `make_videos` to simplify the creation of those after a run.

Since the computations were quite heavy, I chose to implement a *multi-threading* environment to execute all the attacks *concurrently*, thanks to the `threading` library offered by *Python3* language. Moreover, the nodes are removed gradually, in the earlier phases of the attack, until the giant component becomes smaller, I remove a bigger amount of nodes than in the latest ones where I remove one vertex at a time.

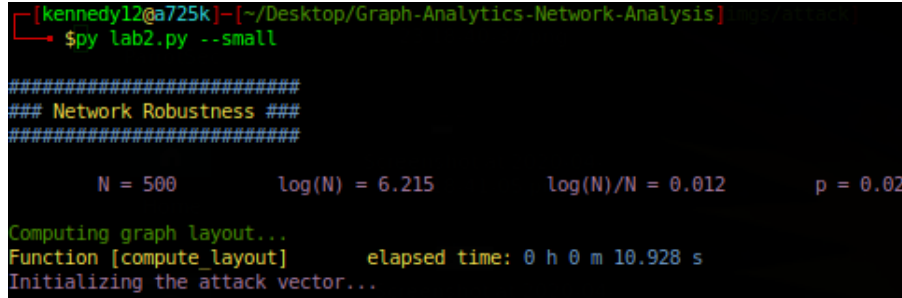


2 Erdős-Renyi Random Graph

To develop the code and to start studying the robustness of a network it is very useful to begin with a small example. In my experiments, I decided to exploit the characteristics of a *random graph*. I did would like to have a **connected** graph, at least before the attacks begin, so I kind of forced this behaviour during the generation phase. In order to achieve this goal, it is important to know the minimum value of p (the probability that two randomly chosen nodes are linked together, i.e. $\exists(i, j) \in E$) such that this condition is satisfied, which is:

$$p > \frac{\log(N)}{N}$$

where N is the number of nodes in the network. This context is also known as the **connected regime**. I chose $N = 500$, $p = 0.02$ to conduct my experiments. In order to trigger this graph generation, it is necessary to launch the main script with the command-line argument `--small` as shown in figure 1.



```
[kennedy12@a725k]~/Desktop/Graph-Analytics-Network-Analysis
$py lab2.py --small

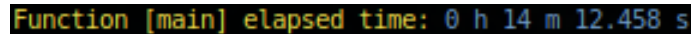
#####
### Network Robustness ###
#####

      N = 500      log(N) = 6.215      log(N)/N = 0.012      p = 0.02

Computing graph layout...
Function [compute_layout] elapsed time: 0 h 0 m 10.928 s
Initializing the attack vector...
```

Figure 1: Statistics of the random graph

Besides its small dimension, the attack required quite some time to terminate, as we can appreciate in figure 2.



```
Function [main] elapsed time: 0 h 14 m 12.458 s
```

Figure 2: Time spent experimenting small random graph

2.1 Random Attack

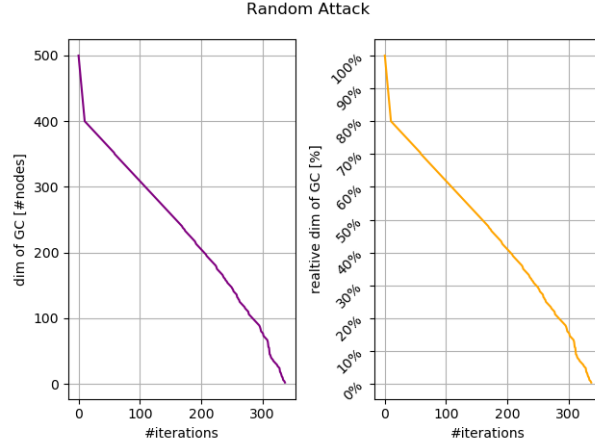


Figure 3: Dimensions of the network under **random failures**

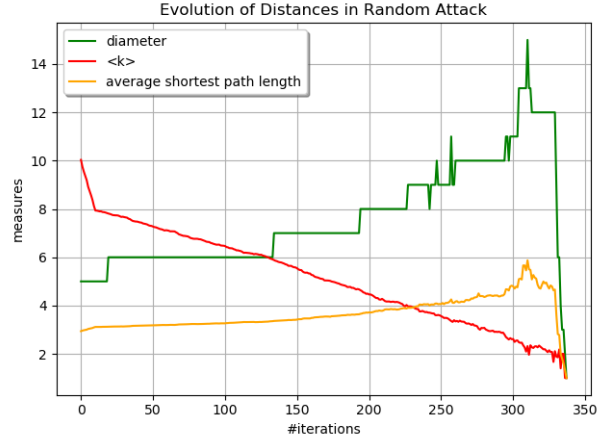


Figure 4: Evolution of the network under **random failures**

In this case, random attacks are very slow, in order to dismantle the giant component, it requires to eliminate more than 300 nodes out of the early 500. The only property that it is actually affected is the *average degree*, $\langle k \rangle$. Notice that the *diameter* of the network grows until a *critical point* after which tends to 0 very rapidly. This fact is observable quite in all the experiments.

2.2 Closeness Oriented Attack

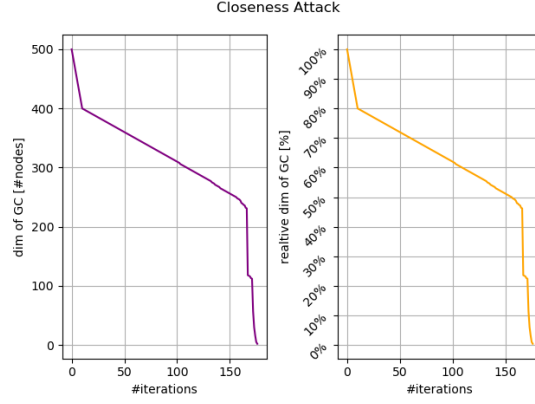


Figure 5: Evolution of the network under **closeness attack**

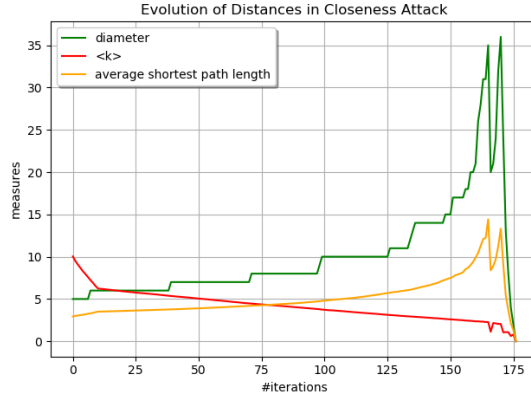


Figure 6: Evolution of the network under **closeness attack**

This attack is more efficient since the number of iterations needed to disaggregate the giant component is approximately half of those counted in random case. Likely because it affects the nodes that, by pure chance got most of the links during the generation phase.

2.3 Betweenness Oriented Attack

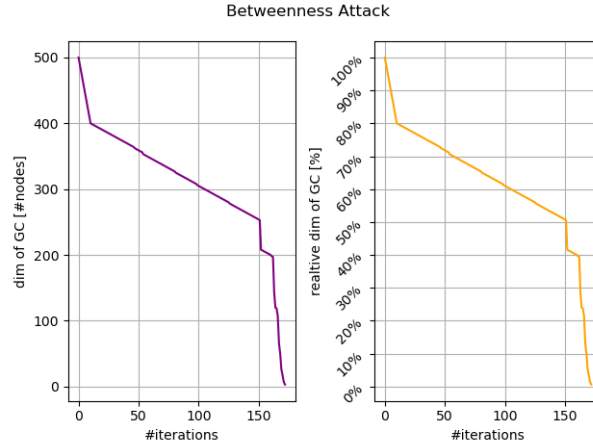


Figure 7: Evolution of the network under **betweenness attack**

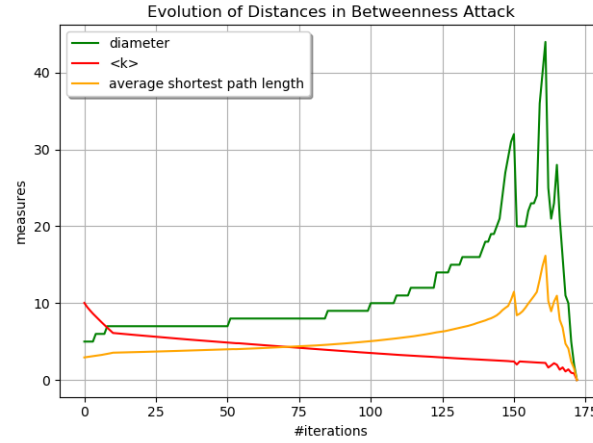


Figure 8: Evolution of the network under **betweenness attack**

In this case, the comments to the previous attack also apply. Observing the plots is evident that this approach is even more powerful, probably because it removes before the nodes not necessarily with the highest degree, but the ones that *glue* together the *components* that have been emerged in the generation phase.

2.4 Hits Oriented Attack

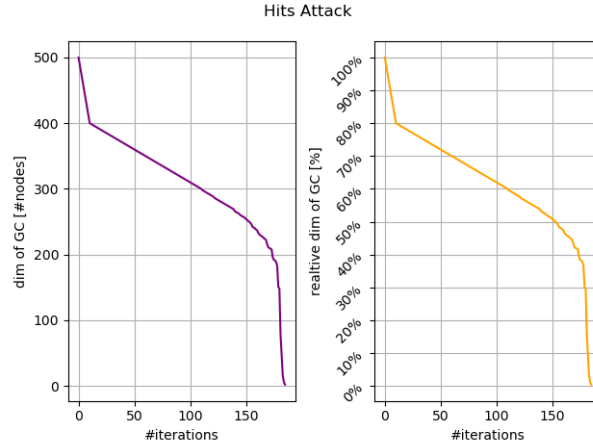


Figure 9: Evolution of the network under **hits** attack

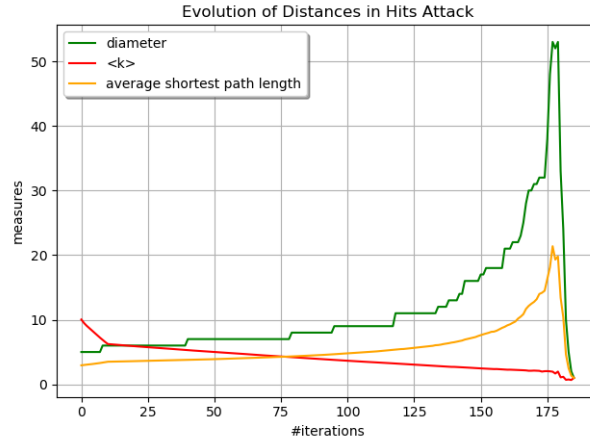


Figure 10: Evolution of the network under **hits** attack

The results for this experiment are a little bit surprising: due to the *random* nature of the network, it is licit to suppose that the presence of a *hub* is not necessarily mandatory. Instead, because this attack converges pretty quickly, I conjecture that there are some nodes in the graph, by chance, with a lot of connectivities, otherwise, this outcome is unexplainable.

2.5 Clustering Oriented Attack

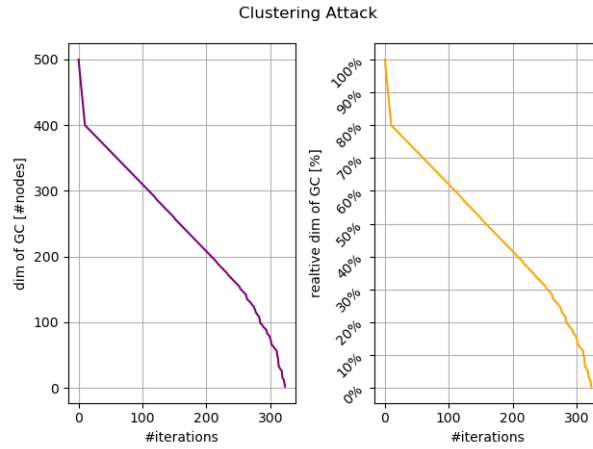


Figure 11: Evolution of the network under **clustering attack**

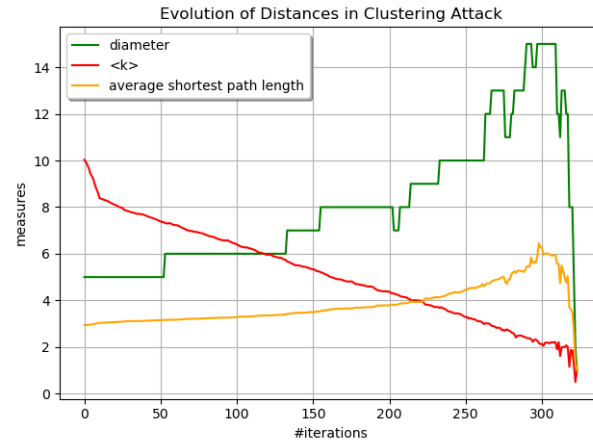


Figure 12: Evolution of the network under **clustering attack**

This kind of attack shows very low performance. Its results are comparable to those obtained in the random case. Not the best metric to choose to harm a random network.

2.6 PageRank Oriented Attack

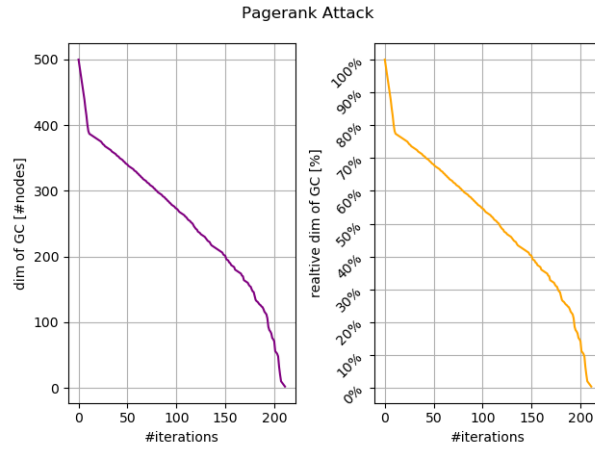


Figure 13: Evolution of the network under **pagerank attack**

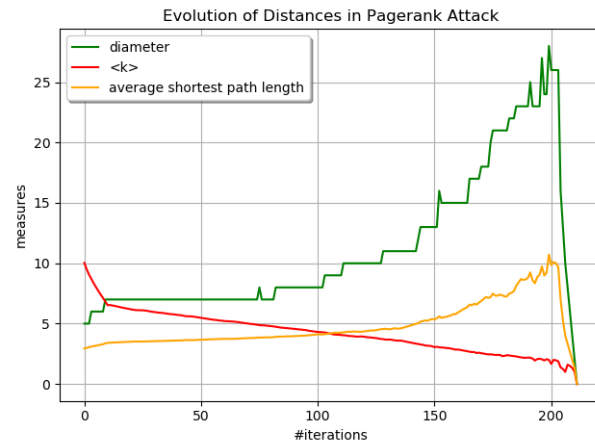


Figure 14: Evolution of the network under **pagerank attack**

This attack, without surprise, proved weak, due to its nature. As this mine the *importance* of a node, it is quite difficult to have some prominent vertex in a randomly generated network. This is in accordance with what I expected in theory.

2.7 Summary

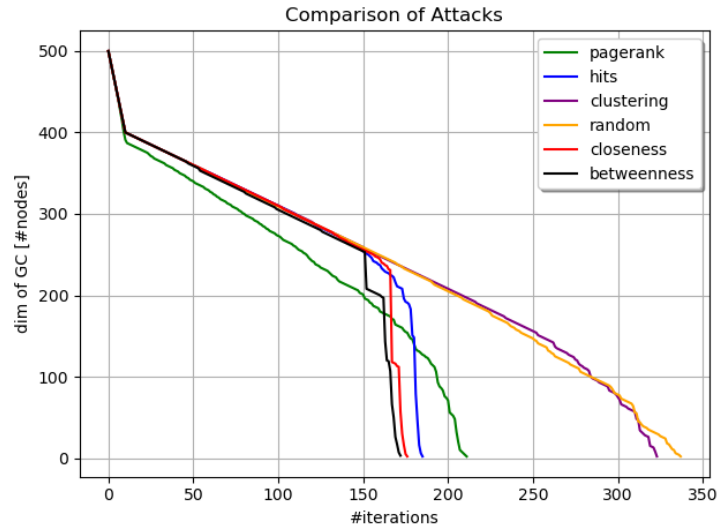


Figure 15: Comparison of all the attacks

Besides *random* and *clustering* attacks performed worse than the others, the difference is not so evident. Whilst the other approaches show similar firepower, the one based on **betweenness** seems to prevail. An interesting point is that the *peak* of the trends is moved to the right w.r.t. the following experiments, indicating again that in random networks no node is more important than the others.

3 American TV Shows Facebook Pages Network

In this section, we can see how an attack could affect a real network. I discussed in the first assignment the *scale-free* nature of the graph, so the expectation is that this network is more *robust* against *random failures* while should be more *susceptible* to *targeted attacks*.

3.1 Random Attack

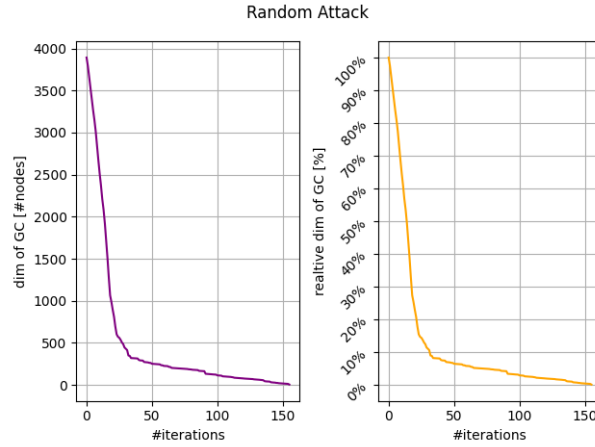


Figure 16: Dimensions of the network under **random failures**

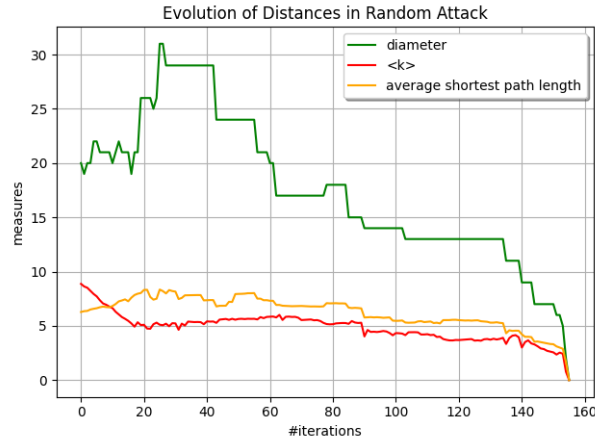


Figure 17: Evolution of the network under **random failures**

According to the graphics, it is evident that, to dismantle the *giant component*, we need to remove the vast majority (somewhat between 75-85% of the total) of the nodes, showing a great capability of the network to cope with *failures*.

3.2 Closeness Oriented Attack

Starting from here and in the following attacks' analysis, we can notice a peculiar trend recurrent in the plots, for what concern the **diameter** of the network. It grows until a peak indicating that the distances in the giant component are augmenting before the unavoidable collapse denoted by the decreasing progress.

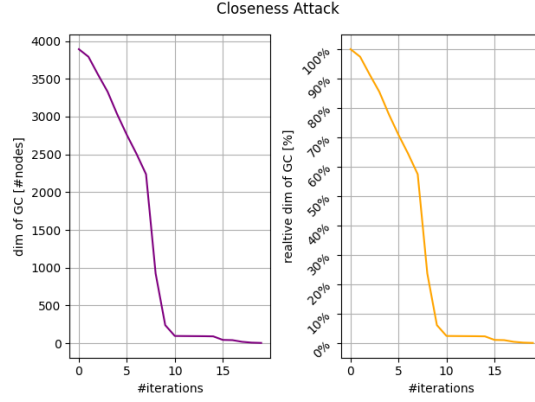


Figure 18: Evolution of the network under **closeness attack**

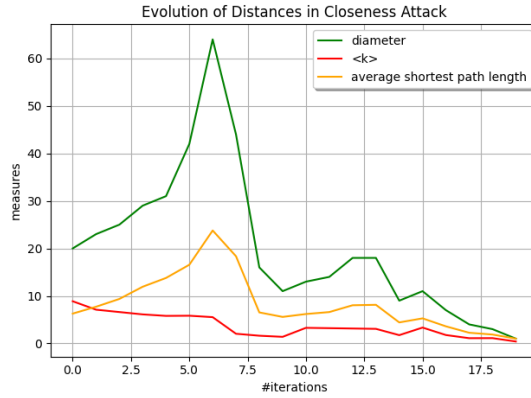


Figure 19: Evolution of the network under **closeness attack**

3.3 Betweenness Oriented Attack

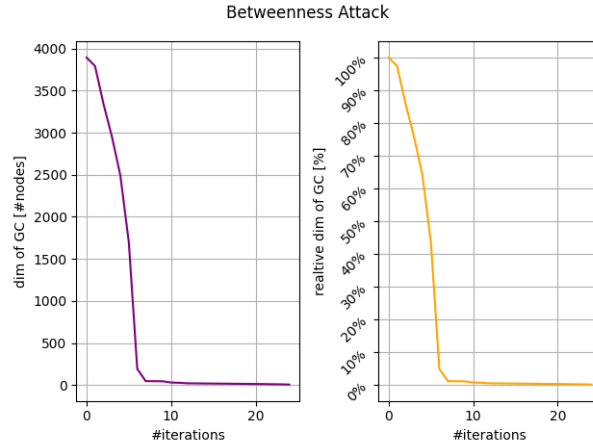


Figure 20: Evolution of the network under **betweenness attack**

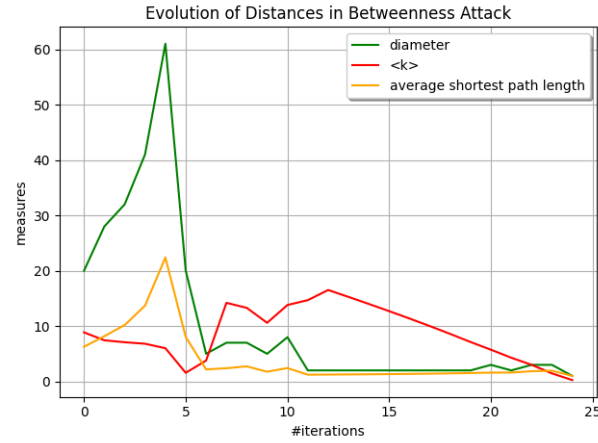


Figure 21: Evolution of the network under **betweenness attack**

Since the peak discussed before is moved to the left w.r.t. the other graphics, I can conclude that this is the *most powerful* attack against the graph. In fact, in less than 10 iterations the dimension of the network size at this point is almost negligible compared to the initial one. Another interesting thing to notice is the trend of the average degree, $\langle k \rangle$, in this experiment.

3.4 Hits Oriented Attack

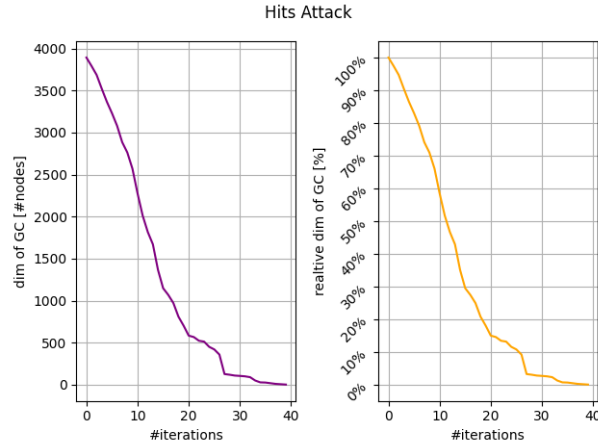


Figure 22: Evolution of the network under **hits attack**

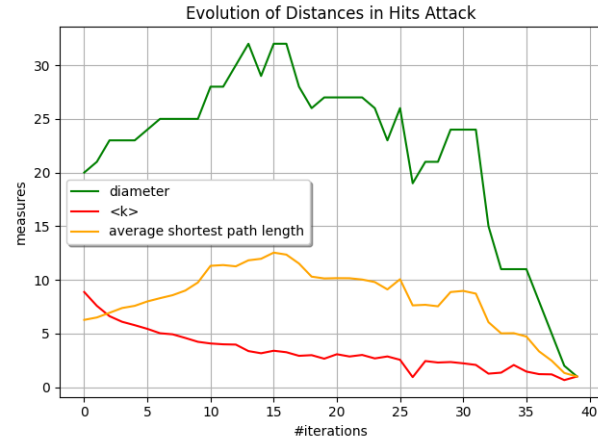


Figure 23: Evolution of the network under **hits attack**

Since the graph is *undirected* so that the values of *hub* and *authority* are the same, I performed only one attack. Unsurprisingly it turned out to be not very effective, nearly all the lines decrease slowly.

3.5 Clustering Oriented Attack

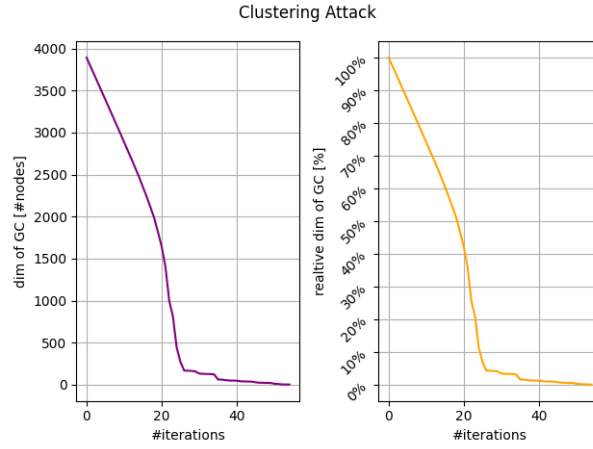


Figure 24: Evolution of the network under **clustering attack**

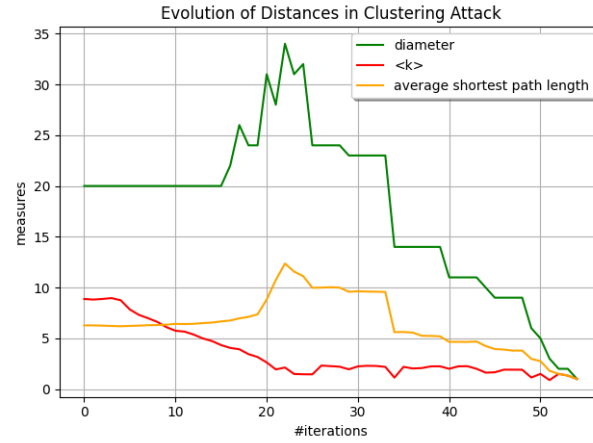


Figure 25: Evolution of the network under **clustering attack**

Similarly to what observed in *hits* attack, here the trend shows the little effectiveness of this kind of approach. The peak is relatively displaced on the right of the picture as good evidence of this fact.

3.6 PageRank Oriented Attack

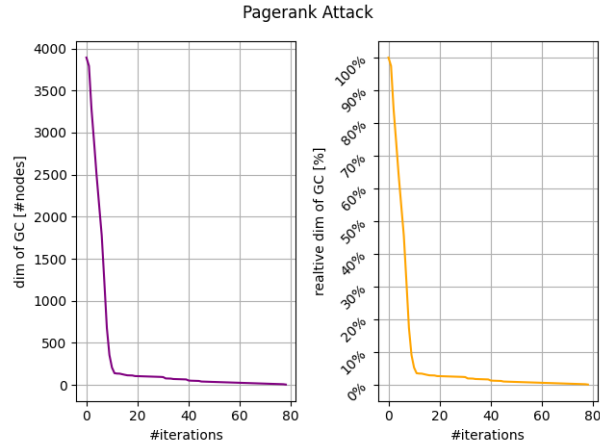


Figure 26: Evolution of the network under **pagerank attack**

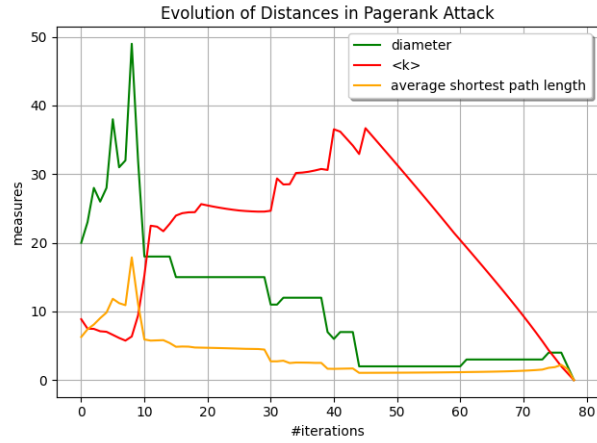


Figure 27: Evolution of the network under **pagerank attack**

This is one of the most interesting trends. There is an early peak but then the curve flattens and in return the average degree increase linearly for quite a bit before collapsing. This is an indicator of the presence of several strong **communities** in the network. Recall that the average degree becomes null when the greatest component eventually vanishes.

3.7 Summary

Without a doubt, this second experiment with a (semi-)real graph was more time and resource consuming. Notwithstanding the effort to optimize the algorithm, for instance removing more than one node at a time, it took really long time, as shown in picture 28.

```
Function [main] elapsed time: 1 h 18 m 8.178 s
```

Figure 28: Time spent on realistic graph experiment

Once again the attack that gives the best performance was the one based on **betweenness**. As expected, *target attacks* converge faster than the random case, thanks to the *scale-free* nature of the network. Compared to the previous experiment, a great result is achieved by the **PageRank**-based onset, especially in the early stages of the attack, even if slow down after.

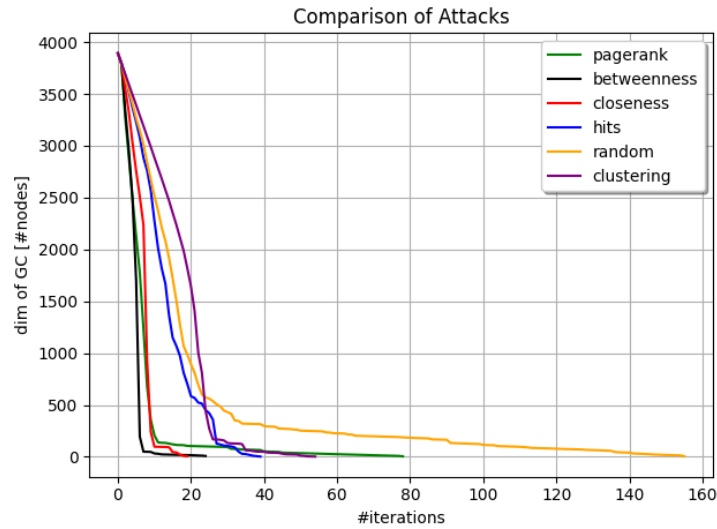


Figure 29: Comparison of all the attacks

References

- [1] Ryan A. Rossi and Nesreen K. Ahmed. The network data repository with interactive graph analytics and visualization. In *AAAI*, 2015.
- [2] Benedek Rozemberczki, Ryan Davies, Rik Sarkar, and Charles Sutton. Gemsec: Graph embedding with self clustering. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2019*, pages 65–72. ACM, 2019.