Percolation in neural networks

Author: Júlia Alberic i Torrent; Advisor: Jordi Soriano Fradera
Facultat de Física, Universitat de Barcelona, Diagonal 645, 08028 Barcelona, Spain.
Departament de Física de la Matèria Condensada.

The study of percolation transitions has proven useful to reveal information of the structure of complex networks, in particular living neuronal networks. Here we considered simulated neuronal networks and use inverse percolation, the process of erasing connections while keeping track of the size of the giant component g, to characterize their resilience to damage. We observed a phase transition in g, revealed by a sudden jump of g at a critical value for the connectivity of the network. We compared the behaviour of different network models (random and scale–free graphs) and different types of attack (damaging connections or neurons, random or targeted attack). We also investigated the critical exponent of the transition for a random graph.

I. INTRODUCTION

With the objective to study how the brain works, and to understand its functionality in relation to its structure (physical connectivity), numerical models of neuronal networks have emerged in the last decade as a powerful modelling tool. These models can explore living networks as mathematical graphs and try to reproduce the functionality of the original circuit.

An interesting concept that this graph approach can explore is percolation. It is a classical problem in statistical mechanics that can be viewed as the transport of information throughout a network. Specifically, it studies if there is a path between two vertices of the graph along which a property (a fluid, information...) could pass. If working with periodic, two dimensional graphs, percolation leads to exact results for the critical point of the transition [1], but arbitrary networks can be studied as well. The appeal of percolation is that it exhibits a phase transition, which means that the system behaves differently in the two regions separated by the critical point. Additionally, the topology of the graph and its statistical traits (average connectivity for instance) are related to this critical point.

When a network is built and studied using percolation, an important property to investigate is the biggest cluster of connected components, called *giant component* g (Fig. 1). It goes up as connectivity c increases (standard percolation) or goes down as connectivity decreases (inverse percolation, the case of Fig. 1). Mathematically, g shows a sudden jump at a critical connectivity c_0 and that separates a connected network from one formed of isolated clusters. This behaviour is characterized as a percolation transition given by a power law:

$$g \sim |1 - \frac{c}{c_0}|^{\beta},$$

where β is the critical exponent. This exponent typically characterizes the topology of the particular network being studied, while c_0 characterizes the average number of connections per node. This means that values of β can be found for different types of network by running simulations, to then potentially compare them to values obtained experimentally in real systems, such as epidemics, rumours spreading, or connectivity in living neuronal networks.

To obtain β, one can plot g vs $|1 - \frac{c}{c_0}|$ in logarithmic scale and fit a power law to get the critical exponent as the slope:

$$ln(g) \sim \beta\, ln\big(|1 - \frac{c}{c_0}|\big).$$

Percolation has been used to study neural cultures experimentally because it gives information on the structure of the network that is difficult to obtain otherwise [2]. Indeed, as illustrated in Fig. 2, the behavior of g as a function of c reveals aspects related to the connectivity of the network, such as the average connections per neuron. In the figure, the blue network has a higher average connectivity than the black one and therefore the critical point c_0 for the blue is smaller. However, both networks have a similar topology, and therefore they have similar β (inset of the figure) [2].

Thus, knowing the importance of percolation in neuronal networks and other systems, we used a computational approach to explore the behavior of different networks. We considered inverse percolation (as in Fig. 1), in which the network gradually breaks down, and investigated also the behavior when either nodes or connections were removed.

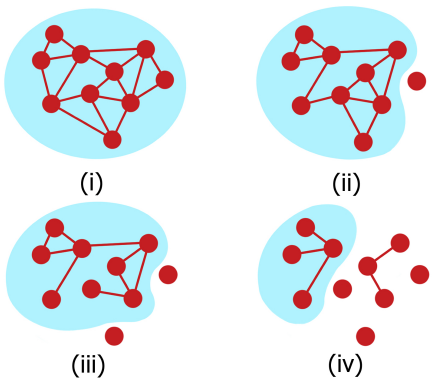

FIG. 1: Sketch of the disintegration of the giant component (blue area) as connectivity decreases from step (i) to (iv).

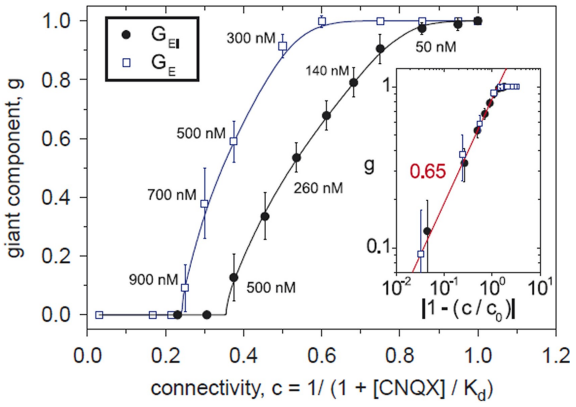

FIG. 2: Example of an experimental percolation transition in *in vitro* neuronal circuits. Main plot: symbols are experimental data in two kinds of networks, highly connected (blue) and moderately connected (black). Inset: log–log plot to obtain the critical exponent. Graph adapted from Ref. [4].

II. METHODS

A. Network generation

As we have said, the critical exponent β is related to the particular topology (degree distribution) in our network. Degree distributions allow us to understand how measures such as node degree —the number of connections each node has with the rest— are distributed across the network. We considered two main cases, single–scale networks and scale–free networks.

Single–scale networks (Fig. 3, left) follow a binomial distribution, where the probability of finding a node with degree k is:

$$P(degree = k) = \binom{N-1}{k} p^k (1-p)^{N-1-k}.$$

Scale–free networks (Fig. 3, right) follow a power law distribution:

$$P(degree = k) \sim k^{-\gamma}.$$

Single–scale networks like Erdős-Rényi graphs or random graph (RG) are used in this study. They have a defined characteristic scale that represents the average connectivity of most nodes in the network. Scale–free networks like Barabási–Albert graphs, also used in our study, do not have a characteristic scale that can be used to estimate the average degree of the nodes. Many real–world networks, such as the brain, are scale–free [3], but neurons grown in a culture are not and follow a random graph [4]. This seems to be related to the lack of functionality of of these cultures. That comparison can be used to our advantage to understand what makes the real brain so capable and highlights the importance of studying these two degree distributions in parallel.

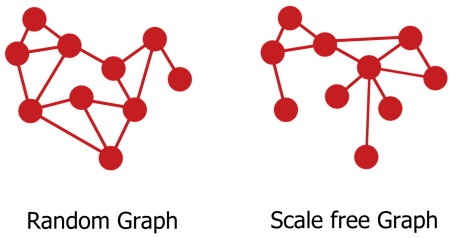

FIG. 3: Example of a random graph and a scale-free graph.

B. Numerical simulations and network analysis

A script was written in Python to make simulations of the evolution of different networks when the connectivity c was reduced. The `NetworkX` package from Python was used to generate both the connectivity (adjacency) matrices that represent the network and visualize them as graphs, taking advantage of the fact that there is an easy conversion between these two objects. The `numpy` package allows us to work with the adjacency matrix, which is a square matrix that represents the equivalent graph.

We mostly worked with networks constituted by 500 neurons. In that case, their adjacency matrices had a dimension of 500×500 nodes, so each component (i, j) has the connectivity information between the neurons i and j. When working with unweighted networks like in our case, the matrix is binary, so a 0 means that there is no connection and a 1 means that there is. The networks used are also all undirected, so the adjacency matrix is symmetric.

Graph objects from the `NetworkX` package were used to extract important parameters using built–in functions, such as the giant component, global efficiency and modularity.

The giant component of a graph g was found by using a `NetworkX` function to extract the size of all the connected paths and selecting the biggest one. The *global efficiency* G_{EFF} was computed directly with a built–in function of that same package and `community` was used to compute the *modularity* Q.

The quantity G_{EFF} quantifies the efficiency of network communication. It varies between 0 for no information flow and 1 for maximum information flow and that corresponds to a scenario in which a node connects with any other. The modularity Q describes the tendency of a network to show their nodes organized in groups, where nodes within a group are more connected with themselves than with the rest of the network. Q varies between 0 (the whole network is the only module) and 1 (each node is independent and shapes by itself a module).

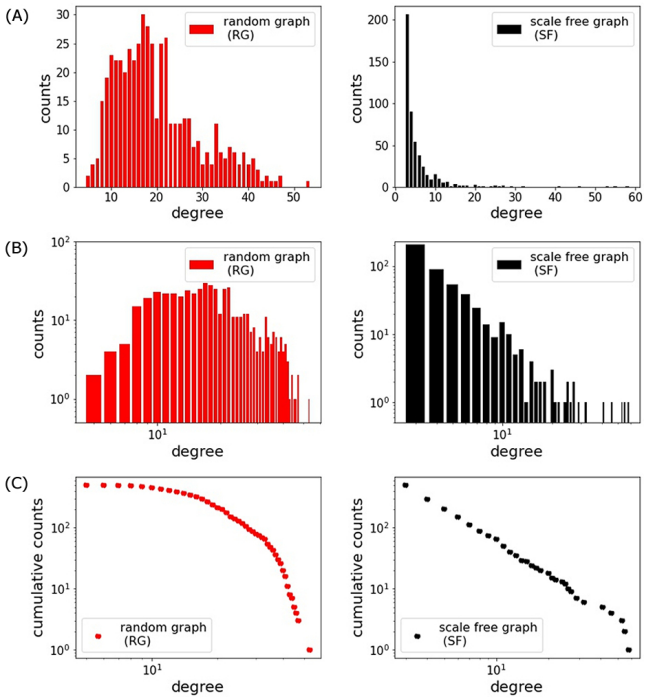

FIG. 4: Degree distributions for the random and scale free graphs used. (A) Histograms depicting the occurrence of a given degree. (B) The same data plotted on logarithmic axes. (C) Cumulative distribution functions, plotted on logarithmic axes

III. RESULTS AND DISCUSSION

A. Networks structure

We worked with two different initial networks; a random one generated by us using the `random` package from Python, in which connections were added to an empty matrix, and a scale–free one generated by the `NetworkX` package using the Barabási–Albert algorithm (degree distribution following a power law). The degree distributions for both are plotted in Fig. 4. We can see the networks behave as expected according to their distribution, which is binomial and power law, respectively [5]. We want to note that these are the initial networks used to be attacked afterwards. To study the networks, algorithms were developed to remove either nodes or edges of these graphs (Fig. 5). Removing nodes was achieved by deleting all the connections of a node (and therefore 'killing' the node). When attacking the network in a random manner, the algorithm randomly selected a node pair (i, j), checked for a connection there, and either eliminated just that connection or the whole node. After each step, the program saved the graph's giant component, global efficiency and modularity. Another algorithm was developed for studying targeted attacks, where neurons with the most connections were eliminated first. In that case, the graph object was used to extract the degree of each node and eliminated the node with the highest degree, meaning the neuron with the most connections, on each step. The program then saved the graph's parameters just like above. The random connection selection was done by using the adjacency matrix and the node selection for both random failure and targeted attack used the graph object.

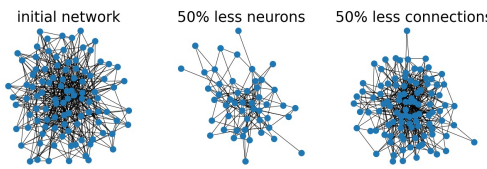

FIG. 5: Graphs illustrating the different cutting methods used in our algorithms.

In the plots that will be presented next, the giant component is the median of ten data sets obtained with different random seeds. The global efficiency and modularity are directly plotted from a single simulation.

B. Inverse percolation on different networks

We know that living neuronal networks are adaptable and repair themselves when they get damaged. For the system to react on time as effectively as possible, it is important that signs of damage become apparent as early as possible. In the opposite scenario, if the living network cannot notice disruptions until the damage has substantially advanced, it will not have time to react. With this ideas in mind, we can judge the robustness and vulnerability of our networks by looking at the shape of the giant component or other curves. If there are more or less flat and suddenly decay, the system is vulnerable, but if they decay gradually, then the system may start response mechanisms.

In Fig. 6A we compare the evolution of the giant component for a network when nodes (red) or edges/connections (blue) are cut. The initial network is the same random graph (RG) made of 500 neurons presented before, for damage actions. Since the initial network has a few connections per neuron, we need more steps in the algorithm in the case of cutting connections to observe the transition, particularly ten times as many steps. Regardless, what we see in the comparison is that the network starts to experience changes when the edges are attacked, since g gradually decays. For nodes, the giant component remains high (whole system connected, although with fewer nodes) until suddenly falls without warning. We see a similar behavior in the global efficiency and modularity plots (Fig. 6B–C), since the curves for node cutting have a steadier behaviour at first and then collapse suddenly. If we think about what is happening, edge removal in a RG changes the network's structure whereas node removal simply causes an equiva-

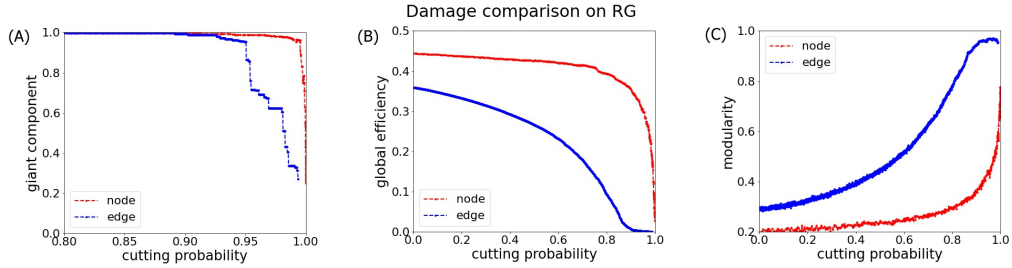

FIG. 6: Network evolution comparison of node versus edge cutting, for our initial random graph with 500 neurons.

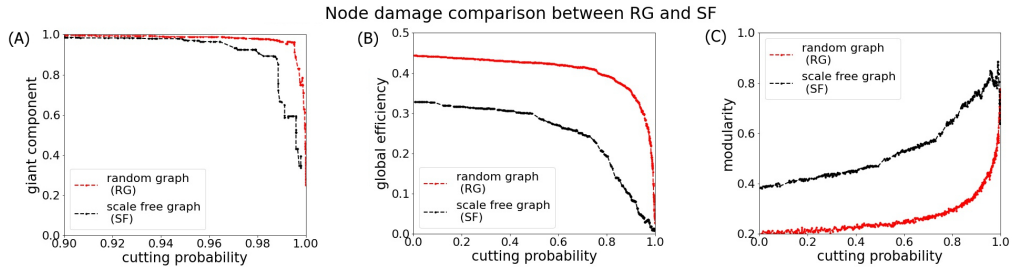

FIG. 7: Network evolution comparison of initial random graph versus scale free graph when cutting nodes.

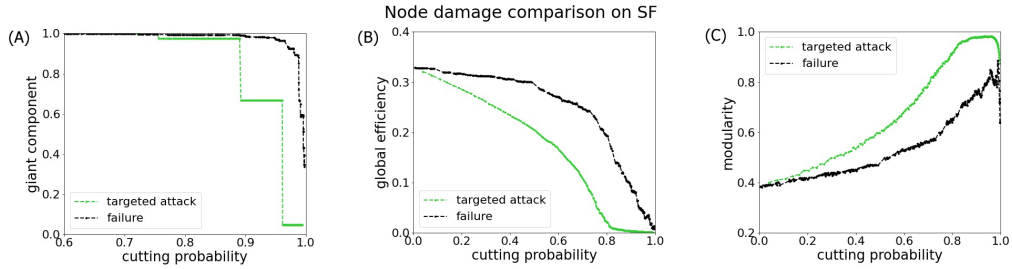

FIG. 8: Network evolution comparison of targeted attack versus failure, for our initial scale free graph with 500 neurons.

lent, but smaller, network to take its place. In this latter case the observed parameters will not change very much until the networks gets really small, as we are observing here. For a living system, it is too late to react.

Next we compared the disintegration curves of two different networks as nodes were cut in an arbitrary order (*random attack* or *failure*). As shown in Fig. 7A, the initial networks are the same random graph (RG) from above and a scale free graph (SF), both made of 500 neurons. In this comparison, an equal number of steps was performed. We can see that the SF decays earlier and gradually since it is possible that a node with many connections is removed. This makes this network more robust from a biological perspective, since it can notice changes and react. The curves for the RG are collapsing later and very suddenly. In this case, we verified that SF are more resilient to random attacks than RG. We can also note the initial values that the global efficiency and modularity take for these two different networks (Fig. 7B–C). RG has higher connectivity and therefore a higher initial value for their global efficiency. RG has a rather uniform degree distribution whereas SF has a modular organization, which makes the modularity higher at the beginning.

Finally, we compared the evolution of a network when cutting nodes either in a random manner or *targeting* in a malicious way, *i.e.*, by deleting first those nodes with the highest connectivity. In this comparison twice as many steps were needed in the random failure case. We used a SF network for this exploration because we know by construction that it is more vulnerable to targeted attacks than a RG, so the difference will be more visible. As we saw previously, scale–free networks have a very different degree distribution than random graphs, and they have few nodes that are highly connected. That means that, if we choose knowingly, we do not have to cut many connections to compromise system's functionality. As shown in Fig. 8A, and as expected, we can see that in the case of targeting the highly connected neurons first (green curve), the network experiences abrupt decays with no previous warning, and where a single jump is almost 50% loss of giant component. SF are thus extremely vulnerable to targeted damage, as expected. The global efficiency and modularity also behave differently

(Fig. 8B–C), changing more abruptly for targeted attack.

C. Percolation exponent

We found interesting to extract the critical exponent for RG graphs and cutting edges, and by using data points near the transition. We run different seeds for the same kind of network to get statistics. Although some data sets had to be discarded, we finally used 7 numerical simulations. As shown in Fig. 9, we plotted g vs $|1 - \frac{c}{c_0}|$ and adjusted a power fit to the logarithmic plot to find an exponent of $\beta = 0.274$. This value is consistent with previous works on percolation in random graphs [10, 11].

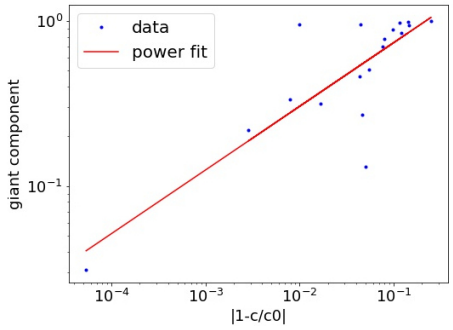

FIG. 9: Power fit of the data points from the transition. Since for each transition we had a whole row of points with the same value of g, only the first one of each was used in this plot for clarity. The value of the critical exponent was found with all the data points.

IV. CONCLUSIONS

Neural networks of sizes 500×500 and 1000×1000 were successfully simulated by using a graph approach in Python, and the desired phase transition was observed. By extracting the giant component, global efficiency and modularity of 500×500 graphs we were able to compare the robustness of two types of networks and cutting methods. The critical exponent β of the transition was found for a random graph of size 1000×1000.

Random graphs (RG) are more robust when attacking their edges because edge removal changes slightly the network's structure, and that can be detected from early on, giving the potential living network time to repair or adapt. Node removal, on the other hand, does not have much of an impact besides making the network smaller but similar, and it is not until the last stages when it is gotten small enough that we can detect the damage.

Scale free graphs (SF) are more robust than random graphs (RG) when removing their nodes [6, 7]. Many biological networks are scale–free because natural selection shapes their formation. After all, different attachments have a certain impact on specimen survival and fitness and an effective preferential attachment can be obtained because of that [8]. Biological networks are in general benefited from this structure because they need robustness to random damage, since that is the type of damage they normally face, for example in the case of mutations in the genome. On the other hand, scale–free graphs (SF) are more vulnerable to targeted attacks than to random failure when removing their nodes. We can see the network losing functionality more suddenly in the targeted attack case, as it is expected for SF. An example of targeted attacks in neuronal networks is Alzheimer's disease, which leaves the brain closer to a RG [9].

Acknowledgments

I want to thank my advisor for his attention and encouragement and my family and friends for their support during this time.

[1] X. Feng, Y. Deng, H. W. J. Blöte. "Percolation transitions in two dimensions" Phys. Rev. E **78** 031136 (2008)
[2] J. Soriano, M. Rodríguez Martínez, T. Tlusty, E. Moses. "Development of input connections in neural cultures" PNAS **105**, 13758-13763 (2008).
[3] V. Eguíluz *et al.*, "Scale-Free Brain Functional Networks" Phys. Rev. E **94** 018102 (2008).
[4] J. Eckmann, O. Feinerman, L. Gruendlinger, E. Moses, J. Soriano, T. Tlusty. "The physics of living neural networks" Physics Reports 0370-1573 (2007)
[5] A. Fornito, A. Zalesky, E. T. Bullmore. *Fundamentals of Brain Network Analysis*, (Academic Press, 2016)
[6] B. Bollobás, O. Riordan. "Robustness and vulnerability of scale-free random graphs" Internet Mathematics **1.1** 1-35 (2004)
[7] A. H. Dekker, B. D. Colbert. "Network robustness and graph topology" Australasian Computer Science Conference **26** (2004)
[8] R. Albert. " Scale-free networks in cell biology" JCS 4947-4957 (2005)
[9] C. J. Stam, W. de Haan, A. Daffertshofer *et al.*. "Graph theoretical analysis of magnetoencephalographic functional connectivity in Alzheimer's disease" Brain **132** 213–224 (2008)
[10] N. Almeira, O. V. Billoni, J. I. Perotti. "Scaling of percolation transitions on Erdös-Rényi networks under centrality-based attacks". Phys. Rev. E **101** 012306 (2020)
[11] Jingfang Fan, Xiaosong Chen. "General clique percolation in random networks" EPL **107** 28005 (2014)