Chapter 1

Centrality Measures in Network Robustness

Contributed by Famim Talukder

1.1 Introduction

Network robustness is a network's ability to withstand failures and perturbations. Human built systems, such as airplanes and power plants, often require maintenance with minor errors, i.e. failure of a single component. However, natural systems exhibit a remarkable ability to retain their principal functions even when they experience failures in multiple components. For instance, in protein-protein-interaction (PPI) networks [3], malformed proteins as a result of bad mutations are common but do not always contribute to diseases. Similarly, in metabolic networks, some chemical reactions are missed, but their consequences are rarely felt [3]. Hence, it is important to understand network robustness and its uses.

Researchers have used network robustness in a variety of fields to further understand systems. In biology, for instance, network robustness is fundamental understanding why some mutations lead to diseases while others go unnoticed. In ecology, it helps determine how the effect of human actions propagate through the environment [27]. In sociology, network robustness is used to determine influence spreaders and decision makers [13]. In engineering, network robustness can determine the weaknesses in modern infrastructure, such as the Internet and power grids [10]. As a whole, network robustness plays a key role in the analysis of system stability and the resiliency they must exhibit in handling perturbations.

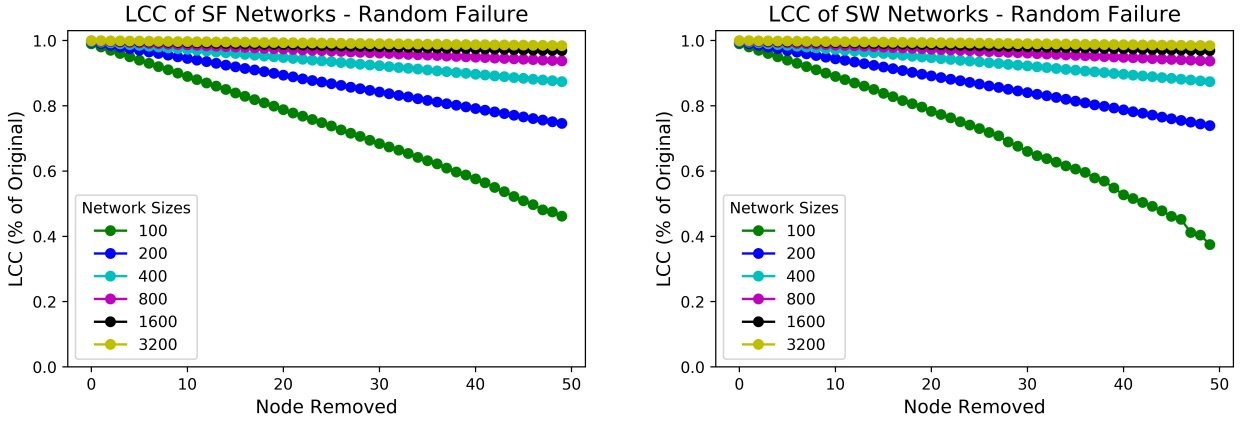
Network robustness extends from percolation theory. Percolation theory discusses the effect on a network where a fraction of nodes or edges are removed. Removal of a few nodes or edges might have a limited effect on the network. However, the removal of several nodes will probably have a more profound effect [3]. Such models are often used to analyze real-world phenomena. The failure of a router or the closure of an airport can be represented as the removal of a node and its edges from its network representation. However, the question then arises: what percentage of the nodes must be removed for the network to lose its functionality? In the Internet example, what percentage of the routers must be nonfunctional for there to not exist any communication between two routers on the Internet? Similarly, how many airport closures will disconnect air travel between two countries? The underlying research question is: how does the disruption of these system affect its overall functionality? To answer these questions, this chapter first delves into the specifics of network robustness.

In this chapter, network robustness is examined examined through its effect on power grids. Power grid networks are comprised of generators, substations and power plants as nodes in the network and edges capture power lines between them. These networks are vital to the economy and safety of the people who rely on them. A failure, such as a severed power line or an electrical fire, can have catastrophic effects on the network. For instance, a blackout involving eleven states and two Canadian provinces in 1996 was a result of a snapped power line in Oregon [3]. As a result, this chapter will explore the robustness and the resiliency of such complex systems.

1.2 The Problem as a Graph

Power grid networks can have numerous different actors, ranging from generators, and substations to power plants. Each of these actors has a different role in the network, and communicate with other actors. Some of the interaction might be directed, while some might be undirected. This chapter only considers undirected edges, where power flow, in this model, is allowed to go both ways. The power grid network is defined as a graph $G = (V, E)$, where $|V|$ is the number of vertices, or nodes, and $|E|$ is the number of edges in our graph. The nodes and edges in these graph can be weighted, i.e. with the probability of failure. However, this chapter studies connected, undirected, and unweighted graphs.

A naïve approach to disconnect power in the network is to randomly remove nodes or edges, to model natural disaster similar to the one in Oregon. Attack strategies relying on removing random nodes will likely not affect the network much, as shown in Figure 1.1, and the disaster originating in Oregon was an unlikely event. However, an attack strategy relying on random removal of nodes will probably require the removal of a large number of nodes, significantly decreasing the potency of the attack. Hence the power grid network is considered resilient to random attacks [9]. A different approach is to remove nodes or edges based on a metric, known as targeted attacks. The removal of a few selected nodes or edges has been shown to deteriorate the network functionality significantly quicker than random removals [23]. In essence, can valuable nodes be determined and removed from these graphs, G, to significantly decrease the connectedness of the network?



(a) Simulation of random failure of a scale-free network.

(b) Simulation of random failure of a small-world network.

Figure 1.1: Simulation of random failure on both scale-free and small-world networks. The largest connected component basically decreases by the removed nodes. The results here are averaged over several runs.

There are numerous different graph metrics used to determine valuable nodes. Degree centrality, for instance, ranks the nodes based on the number of connections or links. Specifically, degree centrality states that the greater the number of connections, the more important the node, whereas a different measure, such as the closeness centrality, ranks nodes based on how close they are to the rest of the nodes in the network. Yet another measure, such as eigenvector centrality, measures how central a node is based on how central its neighbors are [22]. These metrics can be used to rank nodes for a targeted attack and will generally result in different failures. The kernel studied in depth in this chapter is betweenness centrality, a measure which ranks the nodes based on the number of shortest paths between every other nodes which include the node of interest. A formal introduction is presented in section 1.4.4. Betweenness centrality has been proven to be a descriptive measure for flow based networks, like the power grid, which models the flow of electricity [16].

Once these nodes are ranked and removed, the robustness of the network must also be quantified. One basic approach to quantify the robustness is to determine the size of the largest connected component. In power grids, this would quantify the number of generators, substations or power plants that have been disconnected from their source. If the largest connected component is greatly reduced, then major areas experience blackouts as an increasing number of nodes are isolated [14]. A more formal introduction to the largest connected component is presented in section 1.4.5.

1.3 Some Realistic Data Sets

Infrastructure networks are considered to be highly sensitive and, as such, there is only a few openly available data sources. An unweighted power grid network of the western United States containing approximately $5,000$ nodes and $6,500$ edges is used in this analysis [29]. Larger graphs are generated to observe the performance of the sequential and parallel betweenness centrality algorithms. These synthetic networks will be modeled to match power grid data. One such model for power grids would be the Barabási-Albert model, also known as the scale-free model, which are usually used to model the world-wide-web and human chemical reactions. They are generally characterized by a few nodes with high degree and many nodes with very low degree, as such the degree distribution follows a power law distribution [4]. Likewise, a power grid network might have a few nodes of very high degree - a major power plant. There would also be numerous low degree nodes represented by numerous local small town power plants or generators.

Another model which can be used to study power grid networks is the Watts-Strogatz small-world model. This model is generated from a well connected ring lattice, where a number of edges are rewired. The network is characterized by a Poisson degree distribution, large clustering coefficient and low average diameter [29]. Specifically, the length of the shortest chain, l connecting two nodes grows logarithmically with the number of nodes, n, shown in equation 1.1 [5]. The Watts-Strogatz model will be able to capture the relationship between utility poles in a very small town where the diameter, l, grows proportionally with the number of utility poles, n.

$$l \propto log(n) \tag{1.1}$$

Hierarchical models can also used to model power grid networks. Hierarchical networks are scale-free and follow a power law degree distribution, but it also exhibits high clustering [21]. This model will capture the distribution of power from a source to the peripheral, i.e. power is distributed to the small towns after being generated at a major power plant. This model is not studied in this chapter.

1.4 Betweenness - A Key Graph Kernel

Since targeted attacks require targeting specific nodes, measures of node centrality leads to different results. Some common centrality metrics to measure network robustness are presented in the following section. Cudra et al presents a more comprehensive lists of centrality measures used to determine network robustness [14]. The measure implemented and studied in detail in this chapter is the betweenness centrality, covered in depth in section 1.4.4. After the removal of the central nodes based on a metric, an evaluation is done to determine the effectiveness of the metric. This study uses the largest connected component to measure the robustness of a network, introduced in section 1.4.5.

1.4.1 Degree Centrality

One simple but often very apt centrality measure is degree centrality. Degree centrality measures the number of edges incident upon a node. The higher the degree of a node, the more neighbors it has and the more connected it is to the network [16]. Hence, if the graph, $G = (V, E)$, is given in a adjacency matrix A, then the degree centrality of node $i \in G$ is its degree d_i. Specifically, if $n = |V|$, then the degree centrality can be calculated using equation 1.2 and 1.3.

$$A_{ij} = \begin{cases} 1, \text{ if } i \text{ and } j \text{ are connected by an edge} \\ 0, \text{ otherwise} \end{cases} \tag{1.2}$$

$$d_i = \sum_{j=1}^{n} A_{ij} \tag{1.3}$$

Calculating the degree centrality is a fairly simple procedure and requires $\Theta(E)$ to traverse all the edges of G.

1.4.2 Eigenvector Centrality

Another widely used measure is eigenvector centrality which expands upon the degree centrality. Whereas degree centrality measures the direct neighbors of a node, eigenvector centrality gives importance to nodes whose neighbors are themselves important in the network. Specifically, the eigenvector centrality C_e of a node i is defined to be proportional to the sum of the eigenvector centrality of the neighbors of i and is calculated with equation 1.4, where ρ is a constant, $M(i)$ are the neighbors of node i, and $n = |V|$. With a few rearrangements (see reference for the derivation), equation 1.4 can be transformed into the general eigenvector problem, shown in equation 1.5. It should be noted that eigenvector centrality values are all non-negative for a connected graph.

$$C_e(i) = \frac{1}{\rho} \sum_{t \in M(i)} C_e(t) = \frac{1}{\rho} \sum_{j=1}^{n} A_{ij} C_e(j) \tag{1.4}$$

$$AC_e = \lambda C_e \tag{1.5}$$

Eigenvector centrality will basically rank nodes as influential if they are connected to other influential nodes. As such, it can be applied to determine important nodes in numerous real world

applications. For instance, in social networks, a Twitter user who is connected to a few highly influential users, might be more important than an individual who has numerous insignificant followers. A variation of eigenvector centrality is used by Google PageRank and is optimized to handle over 25 billion webpages [2]. Nonetheless, the eigenvector centrality solves an involved equation, which includes solving for the eigenvalues of a graph's adjacency matrix, and the best running time achieved is $\Theta(V^3)$ [20].

1.4.3 k-shell Decomposition

Another metric to determine important nodes in a network is k-shell decomposition. This decomposition starts by removing all nodes of degree 1. The new network is then evaluated and, any node which ends up with a degree of 1, as a result of the removal, is also removed. This procedure is followed until there is no more nodes with degree 1. All node removed will receive a k-shell score of 1. This process is then repeated for all nodes with degree 2 to k, until every node has received a k-shell score [12]. Higher k-shell score corresponds to a more central position in the network.

k-shell decomposition has been used with great success in a variety of different applications. Carmi et al [12] shows that the k-shell decomposition produces insights into the underlying structure of the Internet. Yaveroğlu et al [30] shows that the k-shell decomposition can correctly identify the most influential nodes. Yaveroğlu also shows that the highest k-shell scored nodes do not necessarily have the highest degree, hence the degree centrality and k-shell decomposition produces vastly different rankings.

1.4.4 Betweenness Centrality

Betweenness centrality is common network metric used for various different applications. It has been used to determine interdisciplinary nature of scientific journals [17], information flow between different firms in an alliance network [15], and even evolution of research in collaborative networks [1]. More importantly, it has been used numerous times as the prime centrality metric for determining robustness of power grid networks [11] and communication networks [25].

Betweenness centrality is a global centrality measure based on the shortest paths. This measure considers the number of times a node lies "between" the shortest paths of all other nodes in the network. Specifically, it is defined as the sum of the portion of shortest paths that traverse through the node of interest between the shortest paths of any two other nodes [1]. Formally, the betweenness of a node i is defined in equation 1.6, where σ_{st} is the total number of shortest paths between nodes s and t, and $\sigma_{st}(i)$ is the number of those shortest paths that include node i. Multiple shortest paths between two nodes are permitted.

$$C_B(i) = \sum_{s \neq t \neq i} \frac{\sigma_{st}(i)}{\sigma_{st}} \tag{1.6}$$

Nodes with high betweenness are vital to the the structure and the function of the network. In real networks, these nodes are often associated with power and influence in the organization [6]. In power grid networks, high betweenness centrality will indicate that the node is vital to the performance of the network. Removal of such a node might result in power rerouted to other lines, potentially overloading them. Removal of a significant number of these nodes might cripple the functionality of the network. Since betweenness centrality is significant in flow networks, like power grids, it is studied in depth in the following sections.

1.4.5 Network Robustness Measure

Researchers often use different metrics to quantify the robustness of a network. For instance, the average path length of a network can be used to quantify network robustness, as shown in equation 1.7, where $n = |V|$ and d_{ij} is the shortest distance between node i and node j. The average path length is normalized over all the available node-pairs, $n(n-1)$. A larger average path length means that nodes are farther apart from each other, and removal of a node can significantly increase the average paths between many other nodes, decreasing the robustness of the network.

$$l = \frac{1}{n(n-1)} \sum_{i \neq j} d_{ij} \tag{1.7}$$

Another metric used to quantify robustness is efficiency, E, of a network. Specifically, in power grids and communication networks, efficiency of sending data between two nodes i and j is proportional to the reciprocal of their scalar distance, as shown in equation 1.8, where $n = |V|$ and d_{ij} is the shortest distance between node i and node j. A drop in efficiency, due to a dropped node j, will directly relate to the robustness of the network, often referred to as the efficiency drop, $V_E(i)$. Network robustness efficiency measure can be calculated using equation 1.9, where E is the initial network efficiency, and E_j is the efficiency of the network after the removal of a node. A robust network would have a small drop in the network efficiency with the removal of a node.

$$E = \frac{1}{n(n-1)} \sum_{i \neq j} \frac{1}{d_{ij}} \tag{1.8}$$

$$V_E(i) = \frac{E - E_j}{E} \tag{1.9}$$

In this chapter, network robustness is measured by considering the largest connected component. If removal of a few nodes significantly decrease the size of the largest connected component, then the network is considered vulnerable, i.e. not robust. A study on the robustness of the European Power Grid under targeted attack quantifies robustness by measuring the size of the largest connected component, as shown in equation 1.10, where $n = |V|$ and n' is the number of nodes in the largest connected component. The total run time to compute the largest connected component of a graph is $\Theta(|V| + |E|)$ [24].

$$G = \frac{n'}{n} \tag{1.10}$$

1.5 Prior and Related Work

Network robustness has been studied for infrastructure networks, like power grids and air transport networks. Robustness in such networks guarantee that normal functionality is sustained in the face of failures or attacks. Tu et al [26] studied the robustness of simulated power grid network. These networks were generated to have properties such as scale-free and small world. A variety of different centrality metrics were used. The robustness metric used in this study was the number of unserved stations or, in other words, the number of disconnected nodes. Another study by Wang et al [28] studied the IEEE 57 and IEEE 118 synthetic network power systems using betweenness centrality.

In this study, the robustness measure focused specifically on cascading failures of power grids, failures which would spread throughout the network - a common feature of power grid networks.

Lordan et al [18] studied network robustness in the context of air transport network with betweenness. The study determined that the hub-and-spoke model, which is often used by airlines, is too sensitive to closures and can be easily manipulated. Such designs can have huge financial consequences for airlines in natural disasters, like the eruption of the Icelandic volcano Eyjafjallajökull in 2010, as well as targeted attacks [8].

1.6 A Sequential Algorithm

The state of the art algorithm used to compute the betweenness centrality, covered in section 1.4.4, of a network was developed by Ulrik Brandes in 2001. Brandes was able to reduce the time complexity of betweenness centrality from $\Theta(n^3)$ to $\mathcal{O}(nm)$ and space complexity from $\Theta(n^2)$ to to $\mathcal{O}(n+m)$ [7]. The time complexity is limited by the BFS traversal used to count the number of shortest path, as explained below.

Pseudocode of this algorithm, from Brandes' paper, is provided below. Brandes's algorithm takes advantage of a few observations to achieve this. First, Brandes proves that a vertex $v \in V$ lies on a shortest path, between vertices $s, t \in V$, if and only if $d_G(s,t) = d_G(s,v) + d_G(v,t)$, meaning that the shortest path between two nodes, d_G, that are not neighbors will go through an intermediary node. Moreover, Brandes also proves that if there is exactly one shortest path from $s \in V$ to each $t \in V$, then the dependency, δ, of s on any $v \in V$ obeys equation 1.11, where $P_s(w)$ is the set of predecessors on the shortest path, σ_{sv} is the number of shortest path from s to v. Reduction of this dependency is shown in algorithm 1 line 29.

The very first initialization is a container to hold the results. Next, a stack, queue, and several arrays are also initialized. Arrays σ and d hold the number of shortest paths from the traversed vertices to the current source, s, and the distance of each vertex from s, respectively. $P[w]$, the predecessor set, is a linked-list which contains all the vertices visited prior to v. Lines 9 to 24 is the BFS traversal from the source s, where the distance from the source to each vertex is also computed. For each of the vertices, v, that are found, there is a list of predecessor vertices that are closer to the source, s. Hence, all of these shortest paths must go through its parents and are contained in $\sigma[v]$. Finally, lines 26 to lines 34 computes the betweenness centrality using the dependency.

$$\delta(v) = \sum_{\left(w|v \in P_s(w)\right)} \frac{\sigma_{sv}}{\sigma_{sw}}(1 + \delta_{s|w}) \tag{1.11}$$

1.7 A Reference Sequential Implementation

The sequential algorithm was implemented in NetworkX, an open-source Python library specifically designed to manage graphs. NetworkX provides an easy environment for graphs with many basic graph functions like betweenness centrality measure and the largest connected component. The betweenness centrality function in NetworkX uses an implementation of the Brandes's algorithm, with a run time of $\mathcal{O}(nm)$.

The experiment is designed to compute the betweenness centrality of large graphs of varying sizes and properties. Generators, conveniently provided by NetworkX, are used to produce Barabàsi-Albert scale-free, Watts-Strogatz small-world, and complete graph models of varying

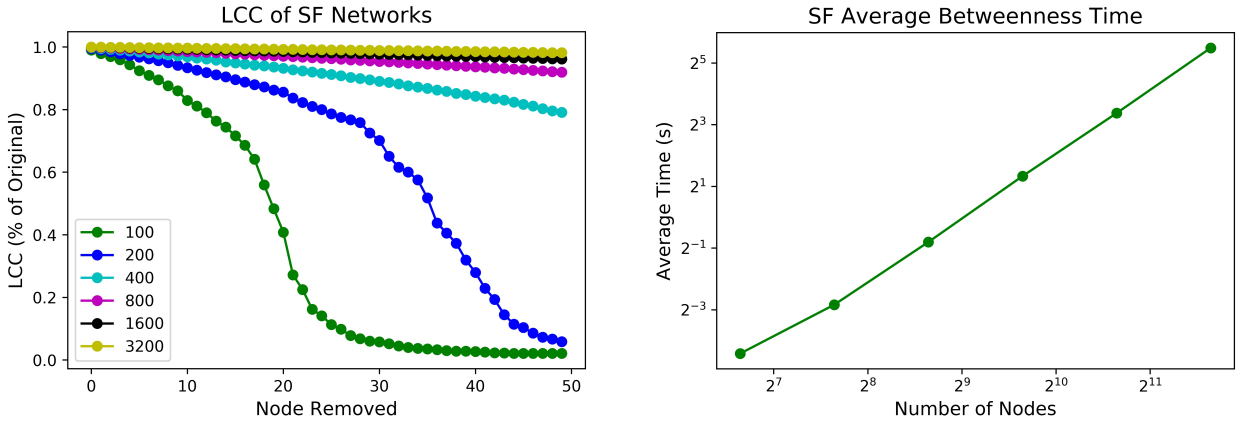
Algorithm 1 Betweenness Centrality in Unweighted Graphs:

1: $C_B[v] \leftarrow 0, v \in V$
2: **for** $s \in V$ **do**
3: $S \leftarrow$ empty stack;
4: $P[w] \leftarrow$ empty list, $w \in V$;
5: $\sigma[t] \leftarrow 0, t \in V$; $\sigma[s] \leftarrow 1$;
6: $d[t] \leftarrow -1, t \in V; d[s] \leftarrow 0$;
7: $Q \leftarrow$ empty queue;
8: enqueue $s \rightarrow Q$;
9: **while** Q not empty **do**
10: dequeue $v \leftarrow Q$;
11: push $v \rightarrow S$;
12: **for all** *neighbor* w *of* v **do**
13: *//w found for the first time?*
14: **if** $d[w] < 0$ **then**
15: enqueue $w \rightarrow Q$;
16: $d[w] \leftarrow d[v] + 1$
17: **end if**
18: **if** $d[w] = d[v] + 1$ **then**
19: $\sigma[w] \leftarrow \sigma[w] + \sigma[v]$;
20: append $v \rightarrow P[w]$;
21: **end if**
22: **end for all**
23: **end while**
24: $\delta[v] \leftarrow 0, v \in V$;
25: *//S returns vertices in increasing order from s*
26: **while** S *not empty* **do**
27: [p[$w \leftarrow S$;
28: **for** $v \in P[w]$ **do**
29: $\delta[v] \leftarrow \delta[v] + \frac{\sigma[v]}{\sigma[w]} \cdot (1 + \delta[w])$;
30: **end for**
31: **if** $w \neq s$ **then**
32: $C_B[w] \leftarrow C_B[w] + \delta[w]$;
33: **end if**
34: **end while**
35: **end for**

sizes. Wall clock is used to time the algorithm as it computes the betweenness measure for all the nodes. Nodes are then removed, in every iteration, based on the largest betweenness value, and all of its edges are disconnected. This experiment allows us to observe the effect of nodes and edges on the runtime of betweenness centrality, as well as the effect of node removal on the runtime and robustness of the network.

1.8 Sequential Scaling Results

The largest connected component is measured to quantify the robustness of the input networks. The results show that for randomly generated scale-free networks, as the graph increases in size, the average time required to calculate betweenness centrality grows super-linearly. Moreover, it also shows that in some cases, with the removal of a handful of nodes (i.e. 16 nodes for the 100 node scale-free model), robustness is decreased significantly, by more than 40% in the 100 node scale-free network.



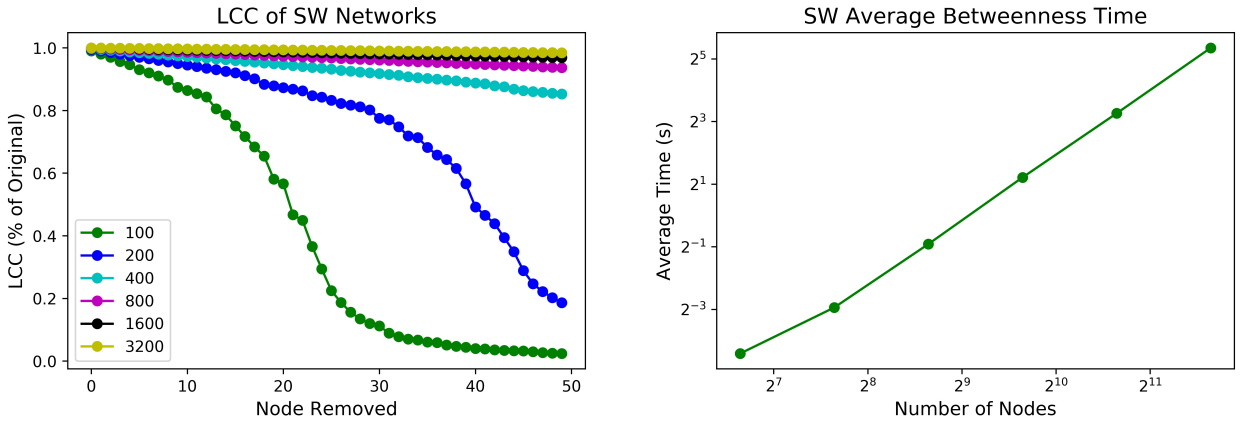
(a) Size of the resulting largest connected component in comparison with the original component with the removal of the highest betweenness centrality node.

(b) Average time required to calculate betweenness centrality, at every iteration, for different size scale-free graphs. Note: the axis are in log-log base 2.

Figure 1.2: Results of scale-free betweenness centrality targeted attacks.

Running the same pipeline on randomly generated small-world networks of varying sizes, produces similar results, as shown in Figure 1.3. The small-world network seems to be slightly more robust than the scale-free network, as removal of 17 nodes lead to a largest connected component of 80%, whereas in the scale-free network the size would be 60%, for the 100 node graph. Moreover, for the 200 node graphs, the small-world network is slightly more robust with a largest connected component size of approximately 85% after 30 node removals, whereas the largest connected component for the scale-free network is slightly less than 80%. Similarly, the average time requirement for the betweenness centrality is also super-linear in the small-world network, and matches the scale-free network very well.

This experiment was also conducted on complete graphs of varying sizes. Complete graphs have all nodes connected to each other, and will contain the maximum number of edges possible, i.e. $|E| = n(n-1) \approx \mathcal{O}(n^2)$. Such a graph would test the upper limit, in terms of edges, of the betweenness centrality since Brandes' algorithm is dependent not only on the number of nodes, but also on the number of edges, hence Brandes' algorithm for complete graphs would have a complexity of $\mathcal{O}(n^3)$. The results are displayed in Figure 1.5, please note that the average time required for the



(a) Size of the resulting largest connected component in comparison with the original component with the removal of the highest betweenness centrality node.

(b) Average time required to calculate betweenness centrality, at every iteration, for different size small-world graphs. Note: the axis are in log-log base 2.

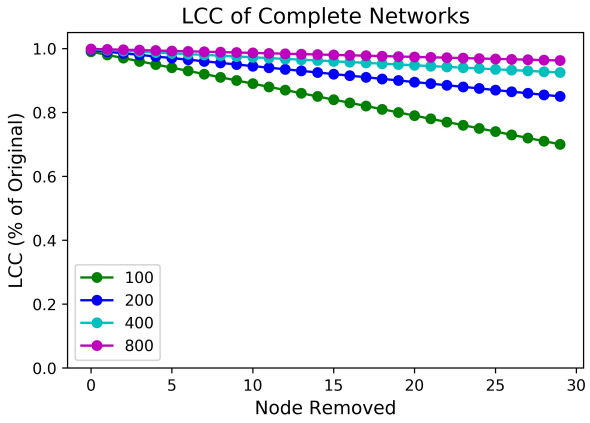
Figure 1.3: Results of small-world betweenness centrality targeted attacks.

800 node complete graph is the same as the time for 3200 node of the scale-free and the small-world networks, showing that complete graphs require significantly more time. Moreover, the complete graph's robustness is the best between the three models, as removal of 30% of the nodes for the 100 node complete graph only decreases the robustness by approximately 30%. This is because in a complete graph every node is connected, so removal of one node does not change the shortest paths between other nodes, since they are all neighbors.

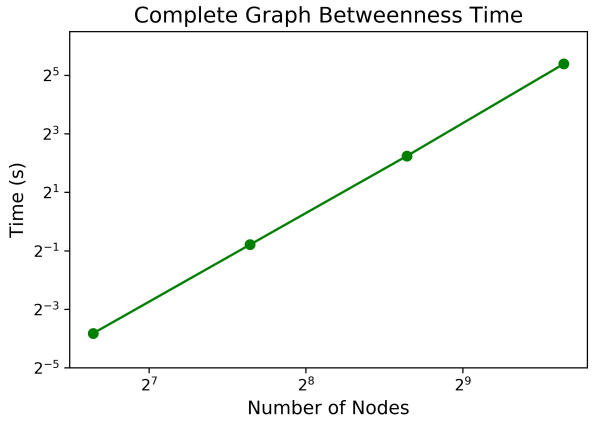
The pipeline was also run on a real power grid network of the western United States. This real data set consists of approximate $5,000$ nodes and $6,500$ edges, with a diameter of 46 edges and the largest degree being 19, but an average degree of 2.67. The low average degree signifies that a large majority of the nodes are connected as an intermediary between two other nodes. However, the maximum degree signifies that there are certain hubs within this network. The degree distribution follows a power law, and hence the graph can be modeled well with a scale-free network. With the removal of just 28 nodes using the betweenness centrality, the largest connected component is decreased by about 35%. In other words, with the removal of just 0.567% of the nodes from the original graph, the largest connected component is decreased significantly. Moreover, as the largest connected component of the network decreases, so does the time required to calculate the betweenness centrality, and is correlated with the size of the largest connected component. It is obvious from this experiment that with the use of betweenness centrality, networks can be attacked successfully, and the largest connected component can be significantly reduced.

1.9 A Parallel Algorithm

Madduri et al [19] presents several different parallel algorithm to calculate betweenness centrality. One approach is optimized to work well on graphs with small diameter. It does so by taking advantage of the sequential Brandes' algorithm and an augmented breadth-first-search (BFS). Each processors execute independently while updating the final centrality score. While the time complexity is comparable to the runtime complexity of Brandes's algorithm, this approach requires $\mathcal{O}(p(n+m))$ memory, where n is the number of nodes, m is the number of edges, and p is the number of processes. The memory constraint make this approach unfeasible on large graphs.

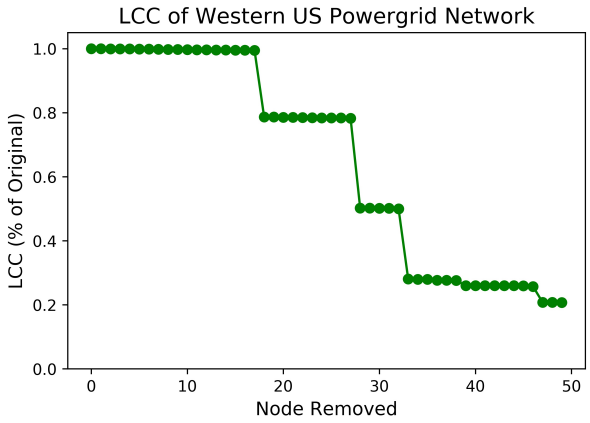


(a) Size of the resulting largest connected component in comparison with the original component with the removal of the highest betweenness centrality node.

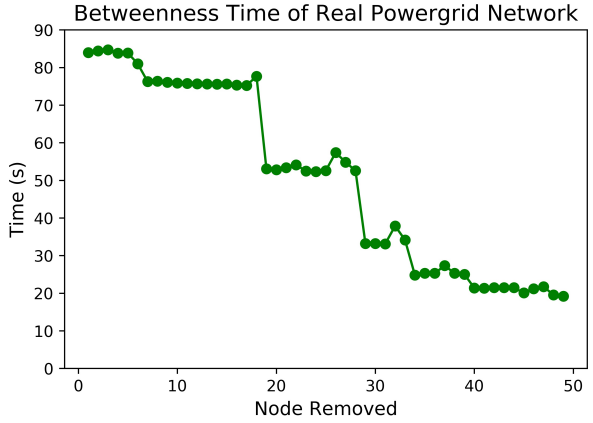


(b) Average time required to calculate betweenness centrality, at every iteration, for different size complete graphs. Note: the size of the graphs differ.

Figure 1.4: Results of complete graph betweenness centrality targeted attacks. The network exhibits very robust properties as the largest connected component for each complete network is exactly 1 less than prior iteration, accounting for the node that was removed. Note that since $m \approx \mathcal{O}(n^2)$, this exhibits the upper bound of time required to calculate betweenness centrality.



(a) Size of the resulting largest connected component in comparison with the original component with the removal of the highest betweenness centrality node. Removal of 28 makes the network vulnerable.



(b) Time required to calculate betweenness centrality at every iteration. Note: the time required to calculate betweenness correlates well with the largest connected component.

Figure 1.5: Results of complete graph betweenness centrality targeted attacks.

The second approach is a fined grained parallelization of augmented BFS. Starting at the source vertex s, The number of visited nodes are slowly increased while simultaneously computing the shortest paths using augmented BFS. A multiset of predecessors associated with each vertex, is maintained, where a vertex v belongs to a multiset of w if $\langle v, w \rangle \in E$ and $d(s, w) = d(s, v) + 1$. The access to the shared data structure, such as the multiset and stack, will need to be synchronized [19]. Using the XMT implementation on 16 processors, Madduri et. al is able to obtain an average speedup of 10.5.

1.10 A Reference Parallel Implementation

A parallel version of the betweenness centrality was implemented using Pool module from Multiprocessing package of Python. The Multiprocessing package supports spawning processing using a convenient API. The parallel implementation uses the fine grained approach, introduced by Madduri et al [19], with augmented BFS. Since Brandes' algorithm is limited by BFS shortest path computation, the parallel implementation reduces the time complexity by a constant factor p, for the number of processes used in parallel.

1.11 Parallel Scaling Results

A thorough performance analysis is conducted on the betweenness centrality implementations on the University of Notre Dame's Center for Research Computing (CRC). A Lenovo NeXtScale nx360 with 12 dual-core 2.5GHz Intel Xeon is used in this analysis. The code is built using Multiprocessing and NetworkX packages in Python. Large input graphs were generated using NetworkX. Barabàsi-Albert scale-free and Watts-Strogatz small-world models were tested in the parallel betweenness centrality. The results, shown in Figure 1.6, display, on average, a 4.3× speedup over the serial implementation. The speedup is skewed, as larger graphs gain larger speedup than smaller graphs.

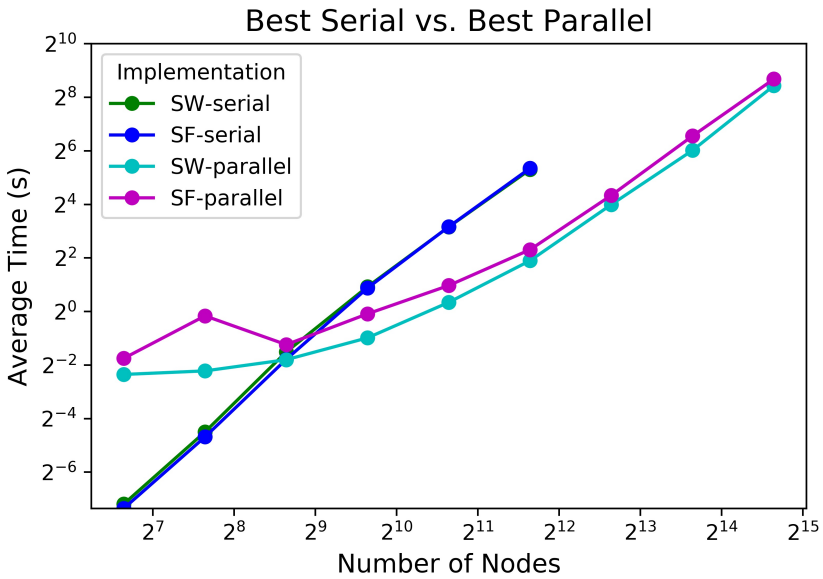
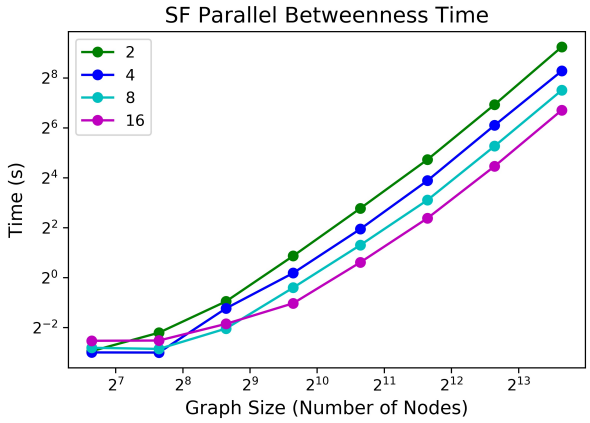
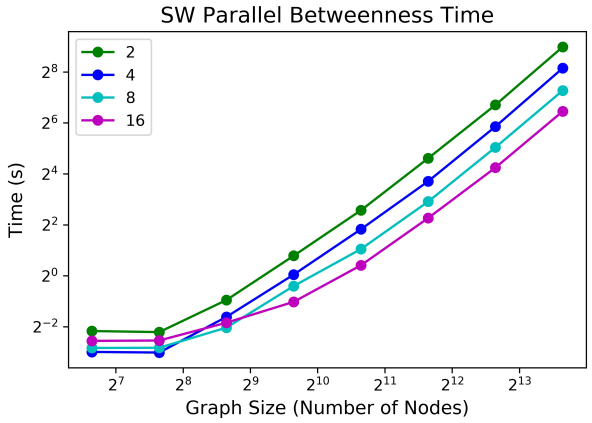


Figure 1.6: The parallel implementation is a constant factor of p faster than the serial implementation of Brandes' algorithm. The Watts-Strogratz model for runs slightly faster for both parallel and serial implementation than the Barabàsi-Albert scale-free model, since the Watts-Strogatz model had fewer edges. The results presented here are for one calculation of betweenness centrality, before removal of any nodes.

The effect of the number of available processes also change the required time of the betweenness calculation. Figure 1.1 shows that as the number of processes double, the calculation time for a graph is halved. The straight line suggests that there is a power law relationship between time graph size and time. Please note that the edges of the networks used in Figure 1.1, are of the order of $\mathcal{O}(n)$, and not $\mathcal{O}(n^2)$, as it is for dense networks. The parallel implementation should produce a steeper slope in a log-log plot, with a slope of 3, since Brandes' algorithm has a time complexity



(a) Effect of parallel betweenness calculation on the scale-free networks. The number of edges are approximately $3 * n$, where n is the number of nodes.



(b) Effect of parallel betweenness calculation on the small-world networks. The number of edges are approximately $2 * n$, where n is the number of nodes.

Figure 1.7: For large networks, doubling the number of available processes decreases the betweenness computation time by half.

of $\mathcal{O}(nm)$, and $|m| \approx \mathcal{O}(n^2)$ for dense graphs. Therefore, it is safe to conclude that the parallel implementation reduces the runtime by a constant factor p for the number of processes allocated to the program.

1.12 Conclusion

This paper discusses the importance of betweenness centrality in the analysis of the robustness of power grids. It is shown that the removal of just 0.6% of nodes, based on the betweenness centrality metric, from a power grid network of the western United States can reduce the largest connected component by approximately 50% of its original size. A sequential implementation of the betweenness centrality in Python using NetworkX is provided. Subsequent parallel implementation using the Multiprocessing and NetworkX packages are also shown, where the parallel implementation shows an average speed up of over 4×, and introduces a constant reduction factor of p for the number of available processes to the program. This speedup is dependent on the number of processes used in the parallel version. The parallel version provides greater speedup for larger graphs, in comparison to speedups of smaller graphs.

There are several areas that can be explored in terms of network robustness as well as betweenness centrality. Betweenness centrality has been shown to be an effective metric in determining the robustness of a power grid. However, there might be better measures, an area I am exploring. In terms of betweenness centrality, work needs to be done to evaluate the runtime of the parallel implementation on dense graphs, which are not representative of power grids and are not explored in this chapter.

Bibliography

[1] Alireza Abbasi, Liaquat Hossain, and Loet Leydesdorff. Betweenness centrality as a driver of preferential attachment in the evolution of research collaboration networks. *Journal of Informetrics*, 6(3):403–412, 2012.

[2] David Austin. How google finds your needle in the webs haystack. *American Mathematical Society Feature Column*, 10:12, 2006.

[3] Albert-László Barabási et al. *Network science*. Cambridge university press, 2016.

[4] Albert-László Barabási, Erzsebet Ravasz, and Tamas Vicsek. Deterministic scale-free networks. *Physica A: Statistical Mechanics and its Applications*, 299(3-4):559–564, 2001.

[5] Alain Barrat and Martin Weigt. On the properties of small-world network models. *The European Physical Journal B-Condensed Matter and Complex Systems*, 13(3):547–560, 2000.

[6] Marc Barthelemy. Betweenness centrality in large complex networks. *The European physical journal B*, 38(2):163–168, 2004.

[7] Ulrik Brandes. A faster algorithm for betweenness centrality. *Journal of mathematical sociology*, 25(2):163–177, 2001.

[8] Peter Brooker. Fear in a handful of dust: aviation and the icelandic volcano. *Significance*, 7(3):112–115, 2010.

[9] Duncan S. Callaway, M. E. J. Newman, Steven H. Strogatz, and Duncan J. Watts. Network robustness and fragility: Percolation on random graphs. *Phys. Rev. Lett.*, 85:5468–5471, Dec 2000.

[10] Duncan S Callaway, Mark EJ Newman, Steven H Strogatz, and Duncan J Watts. Network robustness and fragility: Percolation on random graphs. *Physical review letters*, 85(25):5468, 2000.

[11] Xian-Bin Cao, Chen Hong, Wen-Bo Du, and Jun Zhang. Improving the network robustness against cascading failures by adding links. *Chaos, Solitons & Fractals*, 57:35–40, 2013.

[12] Shai Carmi, Shlomo Havlin, Scott Kirkpatrick, Yuval Shavitt, and Eran Shir. A model of internet topology using k-shell decomposition. *Proceedings of the National Academy of Sciences*, 104(27):11150–11154, 2007.

[13] Meeyoung Cha, Hamed Haddadi, Fabricio Benevenuto, P Krishna Gummadi, et al. Measuring user influence in twitter: The million follower fallacy. *Icwsm*, 10(10-17):30, 2010.

[14] Lucas Cuadra, Sancho Salcedo-Sanz, Javier Del Ser, Silvia Jiménez-Fernández, and Zong Woo Geem. A critical review of robustness in power grids using complex networks concepts. *Energies*, 8(9):9211–9265, 2015.

[15] Victor Gilsing, Bart Nooteboom, Wim Vanhaverbeke, Geert Duysters, and Ad van den Oord. Network embeddedness and the exploration of novel technologies: Technological distance, betweenness centrality and density. *Research policy*, 37(10):1717–1731, 2008.

[16] Swami Iyer, Timothy Killingback, Bala Sundaram, and Zhen Wang. Attack robustness and centrality of complex networks. *PloS one*, 8(4):e59613, 2013.

[17] Loet Leydesdorff. Betweenness centrality as an indicator of the interdisciplinarity of scientific journals. *Journal of the American Society for Information Science and Technology*, 58(9):1303–1319, 2007.

[18] Oriol Lordan, Jose M Sallan, Nuria Escorihuela, and David Gonzalez-Prieto. Robustness of airline route networks. *Physica A: Statistical Mechanics and its Applications*, 445:18–26, 2016.

[19] Kamesh Madduri, David Ediger, Karl Jiang, David A Bader, and Daniel Chavarria-Miranda. A faster parallel algorithm and efficient multithreaded implementations for evaluating betweenness centrality on massive datasets. In *Parallel & Distributed Processing, 2009. IPDPS 2009. IEEE International Symposium on*, pages 1–8. IEEE, 2009.

[20] Natarajan Meghanathan. Use of eigenvector centrality to detect graph isomorphism. *arXiv preprint arXiv:1511.06620*, 2015.

[21] Erzsébet Ravasz and Albert-László Barabási. Hierarchical organization in complex networks. *Physical Review E*, 67(2):026112, 2003.

[22] Britta Ruhnau. Eigenvector-centralitya node-centrality? *Social networks*, 22(4):357–365, 2000.

[23] Sushmita Ruj and Arindam Pal. Analyzing cascading failures in smart grids under random and targeted attacks. In *Advanced Information Networking and Applications (AINA), 2014 IEEE 28th International Conference on*, pages 226–233. IEEE, 2014.

[24] Ricard V Solé, Martí Rosas-Casals, Bernat Corominas-Murtra, and Sergi Valverde. Robustness of the european power grids under intentional attack. *Physical Review E*, 77(2):026102, 2008.

[25] Ali Tizghadam and Alberto Leon-Garcia. Betweenness centrality and resistance distance in communication networks. *IEEE network*, 24(6), 2010.

[26] Haicheng Tu, Yongxiang Xia, Herbert Ho-Ching Iu, and Xi Chen. Optimal robustness in power grids from a network science perspective. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 2018.

[27] Robert E Ulanowicz. Quantitative methods for ecological network analysis. *Computational Biology and Chemistry*, 28(5-6):321–339, 2004.

[28] Xiangrong Wang, Yakup Koç, Robert E Kooij, and Piet Van Mieghem. A network approach for power grid robustness against cascading failures. In *Reliable Networks Design and Modeling (RNDM), 2015 7th International Workshop on*, pages 208–214. IEEE, 2015.

[29] Duncan J Watts and Steven H Strogatz. Collective dynamics of small-worldnetworks. *nature*, 393(6684):440, 1998.

[30] Ömer Nebil Yaveroğlu, Noël Malod-Dognin, Darren Davis, Zoran Levnajic, Vuk Janjic, Rasa Karapandza, Aleksandar Stojmirovic, and Nataša Pržulj. Revealing the hidden language of complex networks. *Scientific reports*, 4:4547, 2014.