CS144: Pandemaniac Report

Ida Huang, Catherine Ma, Amal Tariq and Shen (Irene) Wang

February 24, 2017

1 Implementation

Packages: We used the NetworkX package to create the Graph and to calculate the various centrality measures.

Code Structure: We split our code up into four files: json_parser.py, pick_nodes.py, sim_graph.py, and main.py. json_parser.py contains the parsing code and the code to create the NetworkX Graph object. The main function (in main.py) calls the json_parser.py's parser function to convert the input file into a Graph object and chooses seed nodes by calling the pick_nodes function (in the pick_nodes.py file) with arguments of the graph and the number of seeds needed. The pick_nodes.py file contains a function pick_nodes() that takes an input graph and chooses the seed nodes by calling one of our many strategy functions, which are also in the same file. It also contains an output_nodes function, called by the main function, that prints the seed nodes to a passed output file.

Test Simulations: The sim_graph.py file is used to test various strategies against each other on a particular graph. sim_graph.py takes the seed node output files (generated by the main function through a call to the output_nodes function) and the json input file for the graph. sim_graph.py runs the simulation (using sim.py) and outputs the results to the terminal. The results that are outputted is a dictionary of the strategy name and the number of nodes that were colored by the strategy. This file was used for testing different strategies against each other.

1.1 Design Strategy and Reasoning

We tested various strategies (as seen in the pick_nodes.py file). Our observations for each of the strategies are listed below.

1. **Closeness centrality.** We considered using closeness centrality because choosing seeds that are closest to other nodes (i.e. have the shortest path to most other nodes) would make our strategy able to take over much of the graph quickly. However, we found that for most graphs over half of the top centrality nodes were also the highest degree nodes. When tested against the highest degree strategy, we found that closeness centrality would sometimes win, but the results were not consistent because upwards of 60% of the seed nodes for each competing strategy would be canceled out. So we found that closeness and degree centrality strategies would have similar results.

2. **Degree centrality.** We also tried choosing our seed nodes based on their degree centrality, which is the number of neighbors a node has. We noticed that choosing seed nodes with high degree centralities was quite a strong strategy on a lot of the graphs as we were unable to defeat this strategy without first trying to cancel out its strongest nodes.

   As a strategy against opponents who use degree centrality, we found that using the degree centrality on a portion of our seeds was effective because in the best case, we could use a highest degree node as a seed, and in the worst case, we voided the seed and prevented other competitors from using the highest degree node.

3. **Clustering measures.** We noticed that because we were using closeness centrality, degree centrality and betweenness centrality measures to pick our nodes, we usually ended up picking nodes in clusters that were dense. This is a good approach when we saw that we were competing with teams with the same or fewer than the number of nodes as us, but if there were multiple players then there was a very high probability that the other teams would also

pick these highest centrality nodes which would mean we would have fewer seed nodes. By using clustering, we can figure out which cluster each node is a part of. A good strategy was to ensure that we could maximize our spread over a few of the densest clusters.

4. **Betweenness centrality** Our first attempt to beat the TAs was by choosing nodes with the highest betweenness centrality. The reason we expected this selection of nodes to be a better choice was because we thought that among the top degree nodes the ones with the higher betweenness centralities should span a larger part of the graph and should be able take over more nodes faster. However the betweenness centrality often did not beat other strategies including those based on closeness and degree centrality. We found that this happened because our highest betweenness nodes would overlap with the highest degree nodes, so our results did not improve.

   Additionally, the betweenness centrality was computed too slowly for large graphs. We tried to use a parallel implementation of betweenness to calculate betweenness centrality for the larger graphs but this also took longer than the allotted 5 minutes.

5. **PageRank measures.** The PageRank measure computed the ranking of the nodes in the graph recursively based on the PageRanks of the incoming links. The nodes were then sorted and the top n seeds were used in the strategy, this was done with the heap queue nlargest algorithm. The PageRank measure chose a lot of seed nodes that were also ranked as highest degree, so we tested this method against the highest degree strategy. We found that PageRank often beat the highest degree strategy (in fact, this is how we beat TA-degree, as discussed in the next section). We determined this to be because the nodes that the PageRank strategy selected would first cancel out the strongest nodes selected by the highest degree strategy. The remaining nodes in the PageRank strategy would be connected to other nodes with high PageRanks based on the recursive nature of PageRank, however this was not guaranteed to be the case for the remaining seeds in the highest degree strategy, thus the PageRank method would tend to beat the highest degree strategy.

6. **Weighted combinations.** Overall, we found that the degree centrality and closeness centrality were the strongest strategies in picking the best seeds. We implemented a weighted combination strategy that calculated these centralities for a node and computed the cumulative centralities for the node's neighbors - so we had the node's individual centrality measure and the sum of the node's neighbors' centrality measures. (We also implemented this strategy with additional centrality measures, but found that degree and closeness were the most effective).

   Logically, if the adjacent nodes collectively had a higher degree count, then it would be easier to color more nodes in the graph. The factors were then normalized and a score for each node was calculated by assigning different weights to node degree and node closeness. Two more weights were assigned to the normalized cumulative degree and normalized cumulative closeness of a node's neighbors. The scores of all nodes were then sorted with the heap queue algorithm.

   We experimented with various weights. As stated above, with most graphs, there was significant overlap between the top centrality nodes and the highest degree nodes. The weighted combination implementation was less successful than closeness centrality against other teams. This could have been due to faults in the algorithm and also because the weighted strategy was run on graphs later in the project, so competitors' algorithms likely became more complicated. However, it was able to beat the TA with 20% more seed nodes.

7. **Game theoretic principles.** An option we were considering involved trying to choose nodes that other teams wouldn't choose. One tactic was to pick some popular seeds like highest degree or closest nodes. Choosing said nodes would be effective because they would either become our seed or voided from use by other competitors. Another option we were looking at was to deliberately avoid picking nodes with highest degree, greatest betweenness, and greatest clustering and assume that other teams would pick those seeds and cancel each other out. Then we could choose from any of the other nodes in the graph that were not said seeds.

2 Algorithm Behind Beating Each TA

1. **TA with fewer seed nodes** The closeness centrality strategy defeated this node. Closeness centrality was found to choose seeds with high degrees, thus if the TA with fewer seed nodes chose all the highest degree nodes (which we found to be a pretty strong strategy), our strategy could cancel all or most of the TA's seeds, leaving our closeness nodes which we know would be likely be able to infect the whole graph. If the TA choses a strategy other than highest degree/closeness centrality, then we still know our strategy is likely to succeed since the highest degree and closeness centrality strategies defeat most other centrality measures unless the seed nodes are canceled out.

2. **TA who picks highest degree nodes.** There was an allotted time with which we were able to download the graph being used to compete against the TA. During the set amount of time, we tested all of the above design strategies and found the strategy that would beat the TA with the highest degree nodes on this particular graph. In this case, the PageRank strategy ended up being the one that was able to infect more nodes. The nodes PageRank chose significantly overlapped those chosen by the TA, which worked in our favor because it canceled out said nodes. Our seeds that did not overlap then were able to color more of the graph.

3. **TA who gets 20% more seed nodes.** Our weighted strategy of closeness and degree centrality was able to infect more nodes than this TA. We had tested various weights and determined that the weights would be as in the table below:

contribution	weight
node's degree	1.0
node's closeness	10.0
node's neighbors' degrees	3.0
node's neighbors' closeness	5.0

We deliberately weighted node's degree lower so that we would not pick all of the highest degree nodes as seeds, which we assumed the competing TA would likely choose as their seeds. Node closeness was weighted higher to create a greater gap in importance between degree and closeness. More weight was also put on node's neighbor's degree and closeness so that the seeds chosen would lean toward neighbors with high degree and closeness. By not picking the highest degree and closest nodes, we also avoided having our seed voided. This tactic was aimed at reducing the risk of having our seeds voided while collecting seeds that still had high degree and clustering.

3 Convergence Counterexample

Figure 1 below shows an example of a graph who's colorings do not converge. At every iteration, all the blue nodes will become red because their neighbors are red and all the red nodes will become blue because their neighbors are blue.

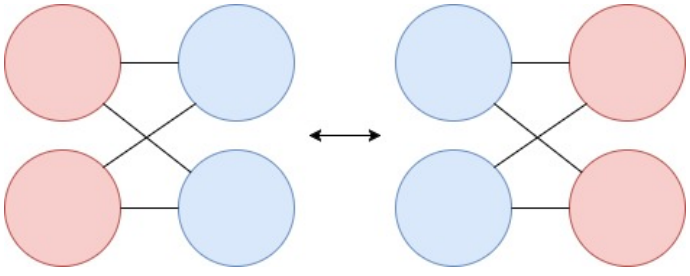

Figure 1: This graph with its initial coloring yields epidemic colors which do not converge

4 Team Contributions

Cathy concentrated on parsing the JSON data files and implemented strategies like the weighted strategy. Amal wrote the structure of the pick nodes file, which Ida, Irene, and Cathy contributed strategies to. Team members individually tested various different strategies and reported their findings to the team. Irene optimized the code and implemented parallel betweenness calculation. Ida tested and contributed strategies to pick nodes. Most programming was done on a single computer, and pair programming was used extensively for the major framework of the code. Each teammate wrote up their relevant section in the report. Amal also wrote up the simulation file for analyzing the various strategies.

5 Suggestions

Some implementations that might make the project more engaging would be a change that will help the teams visualize the graph. Having a picture of which teams won which nodes and what seeds were voided would be very helpful in formulating better strategies during competition. Seeing the coloring of the graph at every round would give the teams insight into how the different algorithms compete against each other.