Progress Report
Week of July 6, 2020

Edris Qarghah

Enrico Fermi Institute
University of Chicago

Weekly Goals

As this is the first week of my internship with Maniac Lab, my focus is on learning about the problems we are currently tackling and working out how I fit into the picture. A few broad categories:

- Get access to the various systems/platforms used.
- Read documentation regarding the platforms we use.
- Review previous work done by members of the team and former interns.
- Explore the existing code-base.
- Study the topics underpinning our work (e.g., network tomography and anomaly detection)
- Develop a toy model so I can begin testing my understanding of these topics.

Daily Log

Monday, July 6

As this was the day before the internship began, I spent some time ensuring I was ready to hit the ground running.

- Followed up with Rob about internship plans/preparations.
- Read about ESNet.
- Reviewed Time Series Analysis and Stochastic Modelling syllabus and supporting materials for references to anomaly detection.
- Gained access to Google Drive folder, Maniac Lab Slack, GitHub repo and personal Google Drive folder.
- Reviewed to-do list.

Tuesday, July 7

- Met with Ilija to discuss Maniac Lab, the internship and related logistics.
- Got access to ElasticSearch, SAND Google Drive and ML platform.
- Set up Skype/Teams.
- Read about PerfSonar.
- Gained access to SAND repo.
- Read work by Suchant.
- Extended Andrew's LaTeX document class and template for weekly reports to match template for SAND presentations.
- Created Google Sheets to track contacts and links to project materials/weekly reports.
- Began first weekly report.
- Spoke with Time Series professor who indicated he would try to work more anomaly detection related material into the course.

Wednesday, July 8

- Further extended weekly report template and updated this week's report.
- Read several resources on Network Tomography.
- Read multiple sources about time series simulation in Python.
- Read overviews of Science DMZ and TCP issues.
- Explored ps_packetloss data in Kibana to get a sense of what I'll be simulating.
- Read several sources about anomaly detection in time series.
- Reviewed several documents from Saul's old stuff, though I'm not sure I got much value from it, as I don't think I have adequate context to understand what it is about.
- Initialized GitHub repo and installed Anaconda.
- Created Jupyter Notebook and tested functionality.

Thursday, July 9

- Researched potential methods and tools for simulating network and time series data, such as SimPy.
- Met with Ilija for clarification regarding toy problem and settled on the details for the project.
  - Build a graph with the following features:
    - The graph has 100 nodes.
    - Each node is a coordinate on the x, y plane.
    - Each node has a random number of connections (up to 4) to its nearest neighbors.
    - The edge/link connecting the nodes have a random bandwidth sampled from 10, 20, 40, 50, 80 and 100 Gbps.
    - The latency for each link is proportional to their geometric distance.
  - Simulate the network:
    - 10 random nodes are selected to "host" PerfSonar.
    - Initial simulation will have a single state, but later iterations will have packet flow that changes over time.
  - Create an observer that will make network measurements.
  - Create a tool that will perform tomography on those measurements.
- Learned of NetworkX from Andrew and read corresponding documentation.
- Researched data-science project conventions and used cookiecutter project template as described in this video.
- Created a graph_test Jupyter notebook that creates a network that meets the minimum requirements discussed with Ilija and displays the network.
- Pushed project, including graph_test notebook, license, requirements, etc.
- Configured compute resource to run the toy network files.

Friday, July 10

- Completed Workday on-boarding tasks.
- Refactored network drawing into a separate function and added saving of figures.
- Created presentation for 2PM meeting.
- Read NetworkX documentation in search of a good method of ensuring trees are not disjoint (i.e, building an MST).
- Set up LaTeXlocally, via MiKTEX and Texmaker, to facilitate pushing to GitHub without having to pay Overleaf.
- Completed weekly progress report and created corresponding makefile.

Achievements

I learned about...

- ESNet
- PerfSonar
- Maniac Lab and SAND
- Network Tomography
- Anomaly Detection
- Science DMZ, common TCP issues
- NetworkX
- Cookiecutter Project template
- Jupyter notebooks

I created...

- This progress report.
- A presentation for our weekly meeting.
- A toy network graph generator.

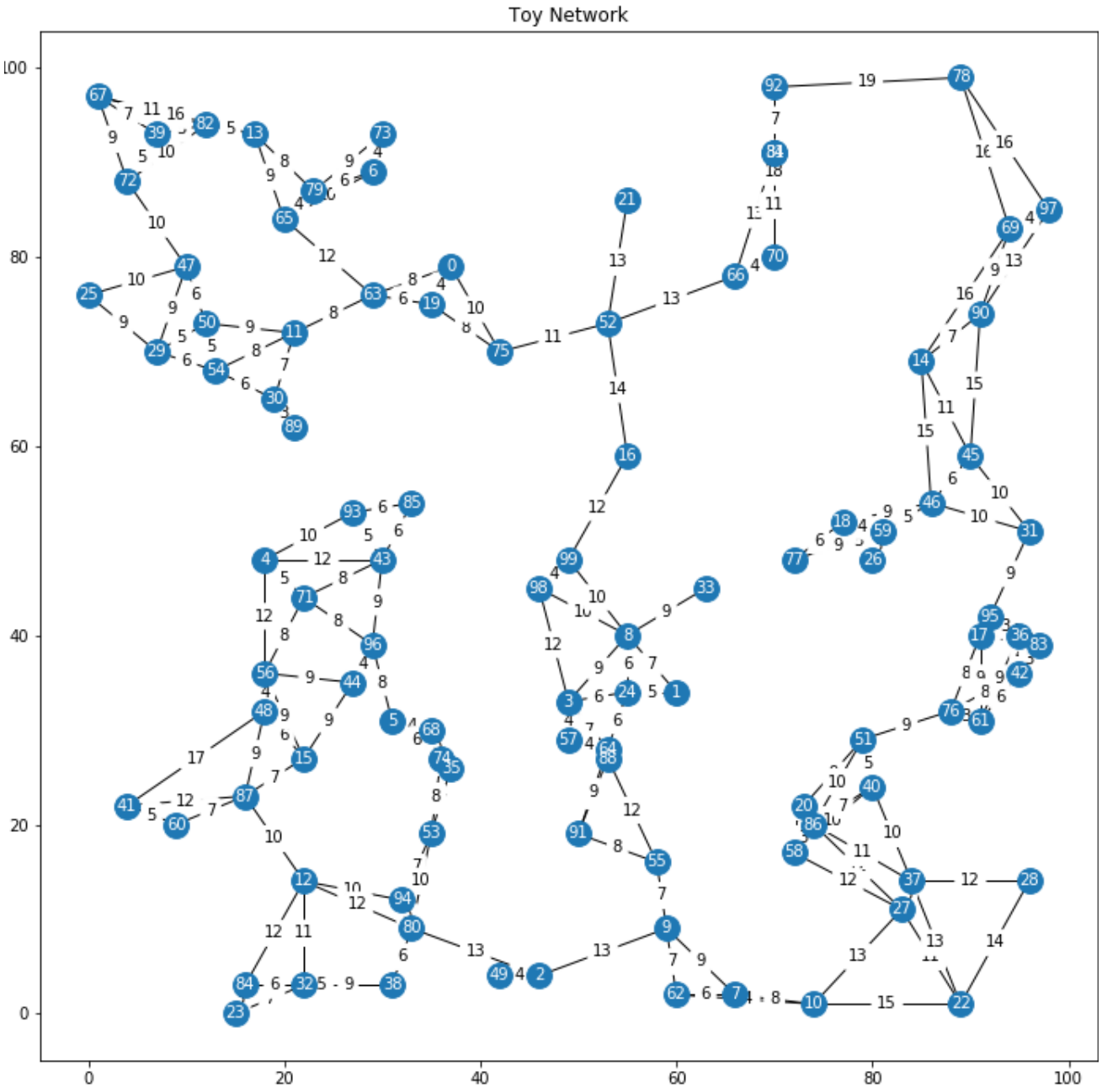

Figure 1: An example of a network I generated.

Roadblocks

Questions

- What are good resources for network tomography?
- Are there good examples for me to use a template for my model?
- What is my title (for resume purposes)?

Problems

Because the criteria for adding edges is to randomly add between 1 and 4 of the closest neighbors to a node, it is possible for a cluster of nodes to all have each other as their closest neighbors. This results in disjoint networks.

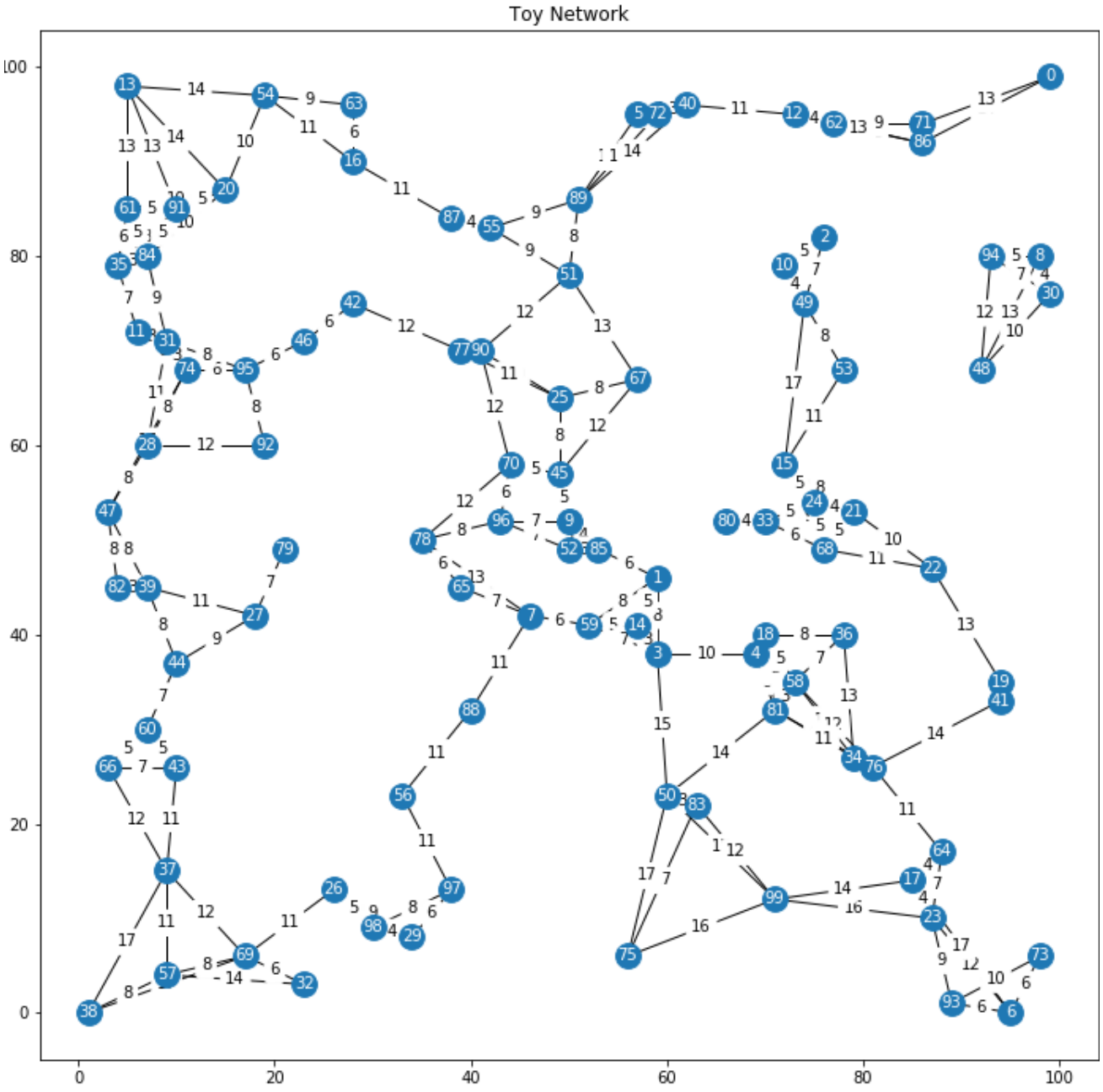

Figure 2: An example of disjoint networks generated.

Challenges

- Displaying graphs on a plane
- Displaying node and edge details

Plans for Next Week

- Toy Network
  - Create a more robust network generation process (e.g., ensuring MST).
  - Simulate network activity.
  - Measure activity from specific nodes and generate a resulting time series.
  - Develop tools for tomography and anomaly detection.
  - Improve visualization of networks.
- Explore real network to understand how to better model it.