Event Driven Network Simulator Report

Challenges and Key Insights

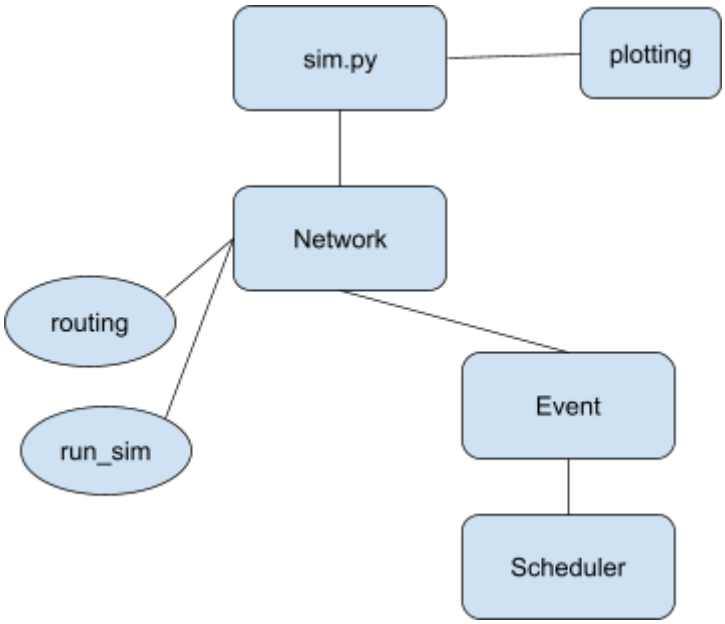
- **Incorporating latency into the Graph:** My biggest challenge I faced was how to add in latencies into the program and I think my code does something with latencies but I am not sure how to evaluate that metric. For now, the packets just accumulate all the weights they encounter on the way from source to destination. I think the ideal way to calculate that would be in units of time but now it is just units. Well, I think that you can think of those units as anything, minutes, seconds, microseconds… but I don't think the graphs would be accurate.
- **Updating Packet Routing:** Adapting the routing mechanism to account for latency, ensuring the simulation uses weighted paths for packet routing. Like for example I was using the shortest path algorithm from the networkx library before adding in latencies and was getting sensible data. But then I added in the mechanism for latencies and then my data would be completely off and I debugged for a long time before realizing shortest path is no longer the best routing algorithm and then I switched to dijkstra algorithm, which accounts for non negative weights in its path and finds the shortest route, and then data started to make sense again.
- **Debugging and Testing**: Ensuring the new features worked correctly within the existing framework, especially given the dynamic and probabilistic nature of network simulations, presented its own set of challenges. I did however started to overcome this challenge by the end when I broke down the entire monolithic file into separate files and suddenly everything felt like it was manageable and I was able to add in rules to possible exceptions.
- **Performance and Optimization: Performance Optimization**: Ensuring the simulation runs efficiently, especially as the network size and number of events increase, posed a challenge. Optimizing graph operations and event handling was crucial for maintaining reasonable execution times. This is a challenge I think I still haven't overcome given that my simulator runs pretty well because the waxman model takes a while to process 1000 events even with a moderately sized graph.
- **Scalability and Extensibility:** Designing the simulation in a way that could be easily extended or modified to include additional features or to simulate different network models and scenarios required thoughtful architecture and coding practices. I don't think my current approach is scalable in any way and I don't

Event Driven Network Simulator Report

think it is doing a good job of working with packets as I am seeing a lot of packets get dropped. not sure if that is my buggy code or if it is how the models are supposed to be. My best guess is I am doing a sub optimal job of routing and latencies

Qualitative Analysis

- Instructions on how to run are in README.md.
- Added in a relatively small unit tests suite but there is work to be done to increase it as the codebase also increases.
- Code has been documented well. Every function has a purpose and every function and variable have appropriate names.
- Git was used extensively throughout the project and timely commits and branches were made.
- I used chatGPT 4 to help me with some of the code writing. If I had to say, I would say like 10-15% of the code is written by chatGPT but the rest of it is my effort and the only open source library I used is networkx library for graphs. I had an initial implementation that didn't depend on it but now my model generation relies on that library because it is simpler but code is modular and the dependency can be removed any time.

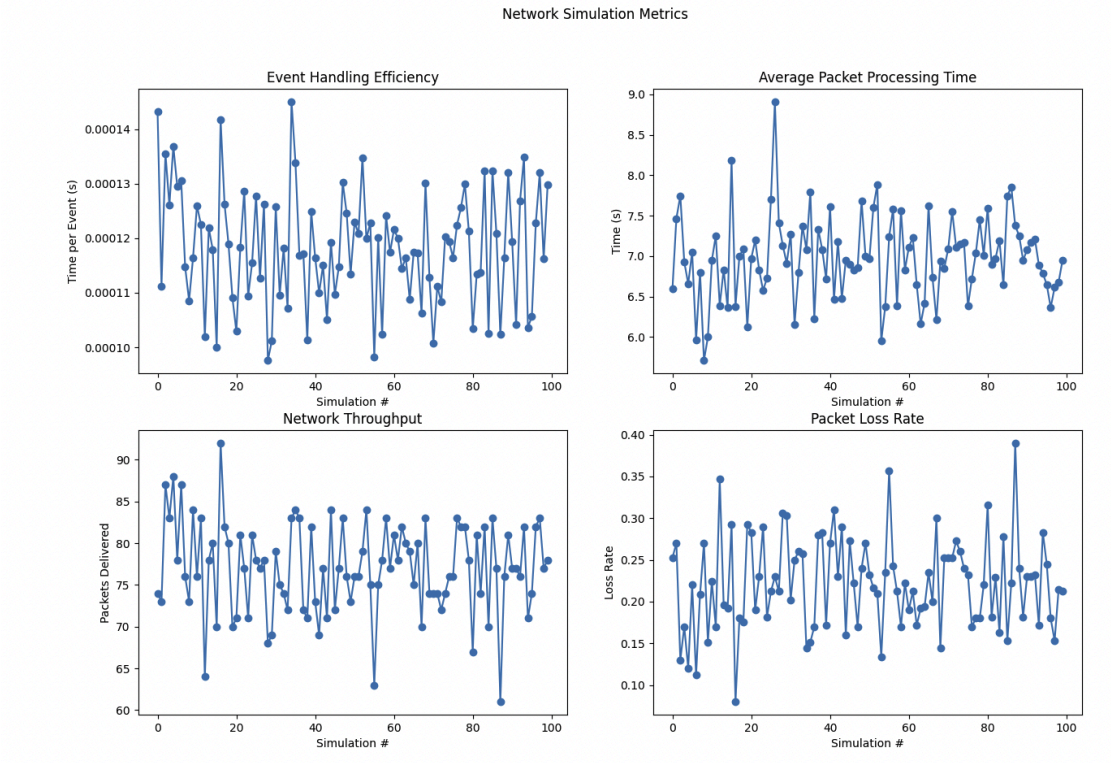


Event Driven Network Simulator Report

Quantitative Analysis

Experiment 1: Barabasi-Albert model with 100 nodes, 500 links, and 100 events.

```
Simulation complete. Success Rate: 0.76, Average Latency: 6.65
Total Latency: 522, Average Latency: 6.365853658536586
Simulation complete. Success Rate: 0.82, Average Latency: 6.37
Total Latency: 549, Average Latency: 6.614457831325301
Simulation complete. Success Rate: 0.85, Average Latency: 6.61
Total Latency: 514, Average Latency: 6.675324675324675
Simulation complete. Success Rate: 0.79, Average Latency: 6.68
Total Latency: 542, Average Latency: 6.948717948717949
Simulation complete. Success Rate: 0.79, Average Latency: 6.95
Latency Distribution Summary:
          Latency
count  78.000000
mean    6.948718
std     2.323776
min     2.000000
25%     5.000000
50%     7.000000
75%     9.000000
max    13.000000
snowcrash:Event-Driven-Simulator/ (mainx) $
```



Event Driven Network Simulator Report

Experiment 2: waxman model with 200 nodes, 1000 links, and 1000 events

```
Total Latency: 3314, Average Latency: 12.007246376811594
Simulation complete. Success Rate: 0.28, Average Latency: 12.01
Total Latency: 4270, Average Latency: 12.270114942528735
Simulation complete. Success Rate: 0.35, Average Latency: 12.27
Total Latency: 4193, Average Latency: 11.745098039215685
Simulation complete. Success Rate: 0.36, Average Latency: 11.75
Latency Distribution Summary:
          Latency
count  357.000000
mean    11.745098
std      5.892722
min      1.000000
25%      8.000000
50%     11.000000
75%     14.000000
max     46.000000
snowcrash:Event-Driven-Simulator/ (mainx) $
```