Computer Networks April 8, 2025

Programming Assignment 3

Due on April 15th, 2025 - 11:59PM *Spring 2025*

1 Overview

Pantheon is a community-driven platform that integrates multiple congestion-control algorithms under a single test harness. Although the official project is no longer accepting new schemes or bug fixes, its open-source codebase is still a valuable reference for understanding how multiple congestion-control algorithms can be wrapped under a unified interface, and to use local network emulation tools (e.g., `mahimahi`) to evaluate algorithm performance under varied network conditions.

In this assignment, you will:

1. Study the Pantheon and MahiMahi papers:
   `https://cs.slu.edu/~esposito/pantheon.pdf`
   `https://cs.slu.edu/~esposito/mahimahi-sigcomm2014.pdf`
2. **Set up** a local testing environment using the existing Pantheon codebase.
3. **Run experiments** with the congestion-control algorithms already implemented in Pantheon.
4. **Measure and analyze** performance metrics such as throughput, latency, loss, etc.

2 Learning Objectives

- **Familiarize** yourself with a real-world congestion-control evaluation framework.
- **Reinforce** your understanding of congestion-control algorithms and their trade-offs in diverse network conditions.
- **Gain experience** using and extending network emulation tools (`mahimahi`).
- **Develop critical analysis** of experimental data to identify algorithmic strengths and weaknesses.

3 Part A: Local Environment Setup

1. Clone Pantheon: Obtain the Pantheon repository: `https://pantheon.stanford.edu/`. `https://github.com/StanfordSNR/pantheon/`
2. Install Dependencies stated in Pantheon's README. Note that you must use Linux or a Linux Virtual Machine
3. Install `mahimahi` for network emulation.
4. Verify example scripts provided by the framework:
   - Run a minimal test with one or two of the provided CC schemes (e.g., TCP Cubic vs. BBR) over an emulated link.
   - Confirm you can collect logs (throughput, RTT, etc.).

4 Part B: Experiments with Multiple Schemes

Pantheon includes or referenced multiple CC schemes (or protocols) e.g., Cubic, BBR, Copa, Vegas, Vivace, etc. Pick and compare 3 protocols. Please be aware that there are tens of protocols supported in Pantheon so it is unlikely that two students will end up comparing exactly the same protocols.

- **Network Profiles:** Use `mahimahi` to emulate two different scenarios:
  1. A *low-latency, high-bandwidth* environment (e.g., 50 Mbps, 10 ms RTT).
  2. A *high-latency, constrained-bandwidth* environment (e.g., 1 Mbps, 200 ms RTT).
- **Running the Tests:**
  - Each experiment should run traffic for at least **60 seconds**.
  - Collect (1) throughput over time, (2) average RTT, and (3) loss rate.
- **Recordkeeping:** Log your data (CSV or parseable text) and commit every experiment you run to your github repository.

5 Part C: Data Analysis and Consideration

Every answer here must be justified by data analysis run in your emulator.

1. **Throughput, Loss, & RTT Comparisons:**
   (a) Plot time-series throughput for each CC scheme to examine ramp-up behaviors.
   (b) Plot time-series losses for each CC scheme to examine behaviors.
   (c) Compare average and 95th-percentile RTT across test scenarios.
   (d) Plot a graph that has all protocols you testbed in the following graph: RTT on the x axis (higher RTT closer to the origin, and lower RTT to the left and throughput on the y axis. Each protocol will have one point on this graph. The best protocol will then end up top-right on this figure.
2. **Identifying Strengths and Weaknesses**
   (a) Which algorithm is more aggressive or latency-friendly?
   (b) Does any scheme persistently overshoot the link capacity?
   (c) Are there excessive losses or large queues with certain parameters?
   (d) Do you have enough information to assess which protocols are the best-performing?

In your report, please repeat the questions before answering.

6 Deliverables (What to submit)

1. **Experimentation Report:**
   - A write-up describing:
     - Your chosen algorithms, network profiles, test methodology (limit 1 page),
     - The report should include all your results: graphs (no tables) of throughput, latency, and any additional stats (no page limits).
2. **Code/Script Submission:**
   - Submit any custom scripts (Python, shell, etc.) that you used to run and parse Pantheon experiments.
   - Include clear instructions for replication on a standard Linux environment.

7 Evaluation and Grading Criteria

- **Data Analysis & Discussion (80%)**
  1. All graphs requested are submitted
  2. All protocols requested are compared
  3. The README file is complete and the experiments are reproducible (graders can replicate all results easily).
  4. Clear, well-labeled graphs, strong comparative analysis.
  5. Each graph is explained with a what (what it is) and a so what? (what do you think the result means)
  6. The code that is used to run Pantheon with Mahimahi and the code used to generate all graphs is clear and complete.
- **Lesson learned (20%)**
  1. What was the most challenging part of this assignment?
  2. If you used LLMs, explain how. What do you think you were going to learn better without LLMs and what do you think helped you learn faster?
  3. With whom (human e.g., other students, TA) did you discuss your solutions?

8 Submission Instructions

Submit a PDF of your report on Canvas. The report must include a Git repository link with your scripts, code modifications, data, and instructions on how to reproduce your solution. The repository should be public so the grader can access it.