BLG 433E Computer Communications Project 4

Due: Tuesday, May 28, 2024

Lecturer: Prof. Abdul Halim Zaim CRN: 21358

Fatih Baskın
150210710

Contents

Ethernet Protocols

Pure Aloha

Pure Aloha is a simple network protocol for multiple access channels where each device sends data whenever it has data to send, without checking if the channel is free.

- Devices transmit data packets whenever they have data to send.
- If a packet collides with another packet, the sending device waits a random amount of time before retransmitting.
- Collisions are detected through the lack of acknowledgment from the receiver.
- Efficiency is low, with a maximum throughput of 18.4%.

Slotted Aloha

Slotted Aloha is an improvement over Pure Aloha, where the time is divided into discrete slots and devices can only send data at the beginning of these time slots.

- Time is divided into equal-sized slots, synchronized across all devices.
- Devices can only transmit at the start of a time slot, reducing the chance of collisions.
- If a collision occurs, the device waits for a random number of slots before retransmitting.

Persistent CSMA (Carrier Sense Multiple Access)

Persistent CSMA is a network protocol where a device first listens to the channel before transmitting and continues to listen if the channel is busy.

- The device listens to the channel to check if it is free (carrier sensing).
- If the channel is free, the device transmits immediately.
- If the channel is busy, the device continues to listen persistently until it becomes free, and then transmits immediately.
- This approach reduces collisions compared to Aloha protocols but can lead to increased delays and inefficiencies in high traffic.

CSMA/CD (Carrier Sense Multiple Access with Collision Detection)

CSMA/CD is a network protocol used in Ethernet networks to improve upon CSMA by detecting collisions during transmission and managing retransmissions.

- The device listens to the channel to check if it is free.
- If the channel is free, the device starts transmitting.
- The device continues to monitor the channel while transmitting.
- If a collision is detected (multiple devices transmitting simultaneously), the device stops transmitting.
- All colliding devices wait for a random backoff time before attempting to retransmit.
- CSMA/CD is effective in wired networks where collision detection is feasible, like Ethernet.

Simulation Code

Setup

To run the `simulation.py` script, you need to create a virtual environment, install the required packages from `requirements.txt`, and then run the script within the virtual environment. First, create a virtual environment by navigating to your project directory in the terminal and executing `python -m venv env`. This will create a new virtual environment in a directory named `env`. Next, activate the virtual environment. On Windows, use `env\Scripts\activate`, and on macOS and Linux, use `.  env/bin/activate`. With the virtual environment activated, install the required packages by running `pip install -r requirements.txt`. After the installation is complete, you can run the `simulation.py` script within the virtual environment using `python simulation.py`.

Simulation Explained

`simulation.py` conducts a detailed simulation of multiple access protocols: Pure Aloha, Slotted Aloha, Persistent CSMA, and CSMA/CD.

Within the Simulation class, each protocol is simulated individually using sophisticated mathematical calculations. These calculations involve probability distributions, such as Poisson distributions, to model packet generation and transmission. For instance, the number of packets generated by each device follows a Poisson distribution with a mean corresponding to the network load.

Note that this is not a real simulation, just a mathematical model. It has several inaccuracies such as throughputs of CSMA protocols dying off after G goes to 1. It should get closer to 1.

The results of the simulations are saved as `.png` files for each simulation. Simulation occurs for Pure Aloha, Slotted Aloha, 0.5-persistent CSMA, 1-persistent CSMA, and CSMA/CD. The number of devices are simulated for 5 to 50 devices, in iterations of 5.

The simulation will yield 10 `.png` files and `simulation.log` file. I didn't want to clutter the console therefore logging is done separately into a file.

Simulation Results

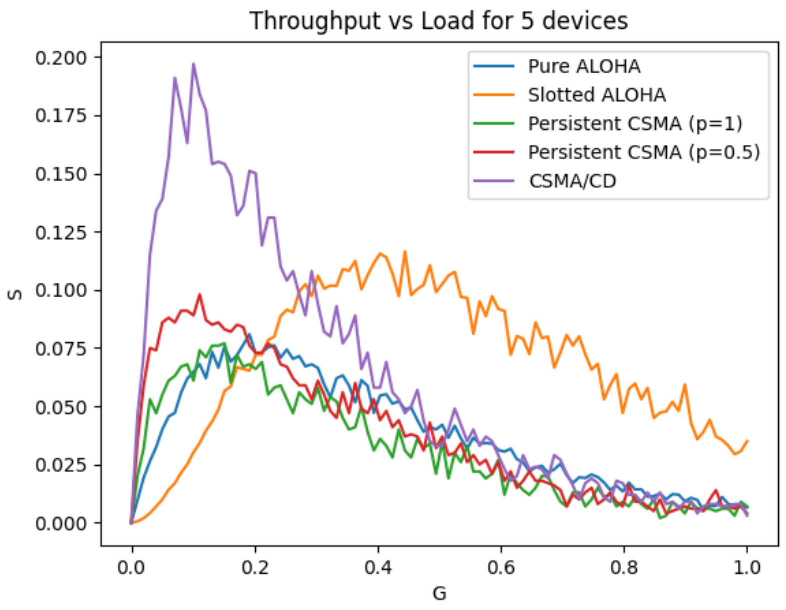

Figure 1: Simulation results for 5 devices

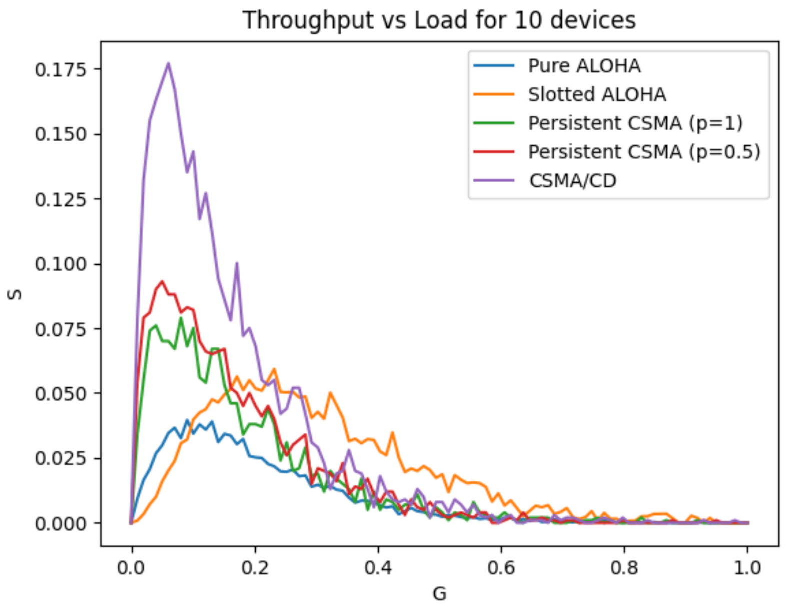

Figure 2: Simulation results for 10 devices

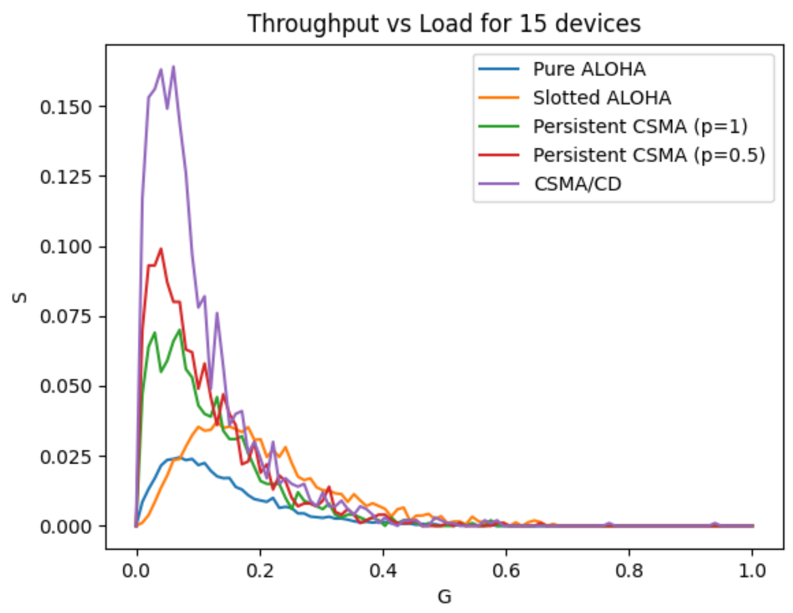

Figure 3: Simulation results for 15 devices

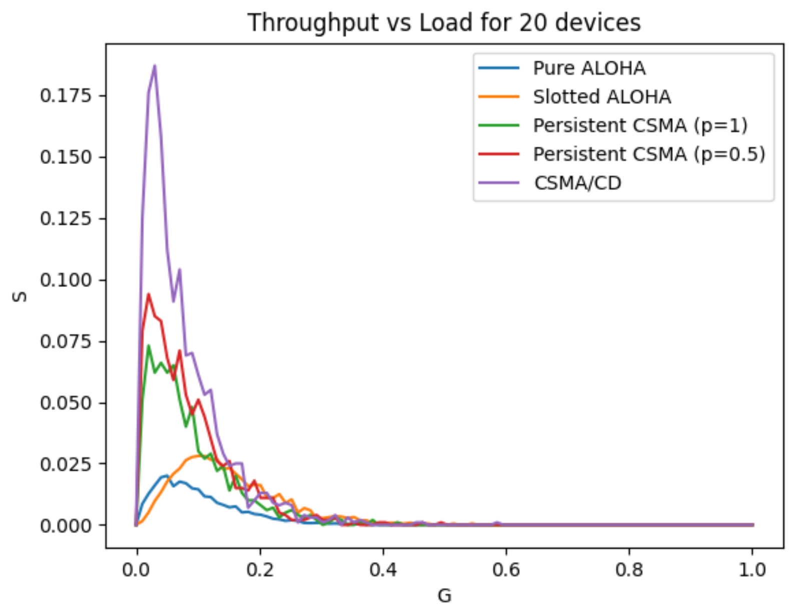

Figure 4: Simulation results for 20 devices

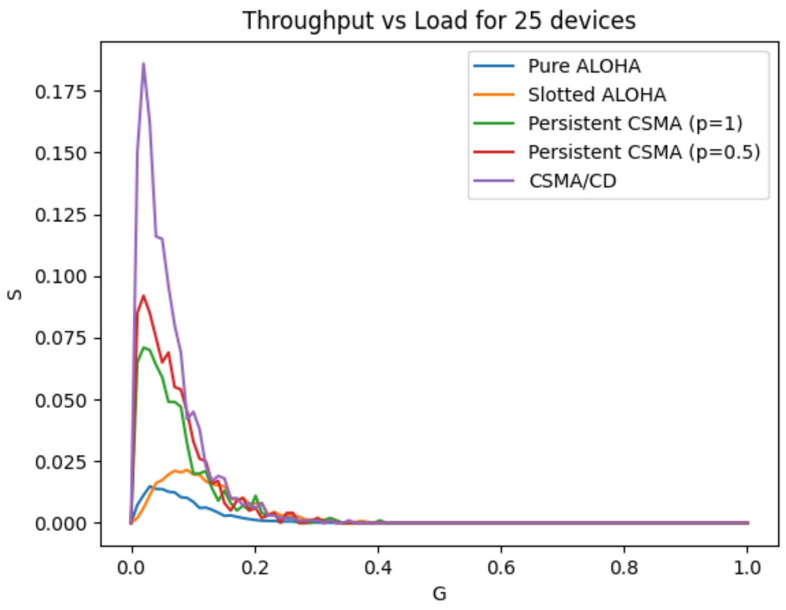

Figure 5: Simulation results for 25 devices

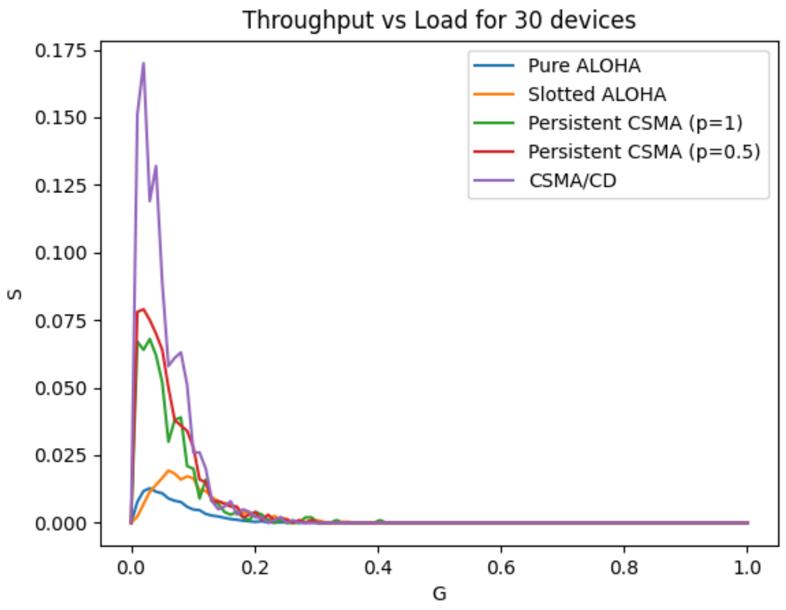

Figure 6: Simulation results for 30 devices

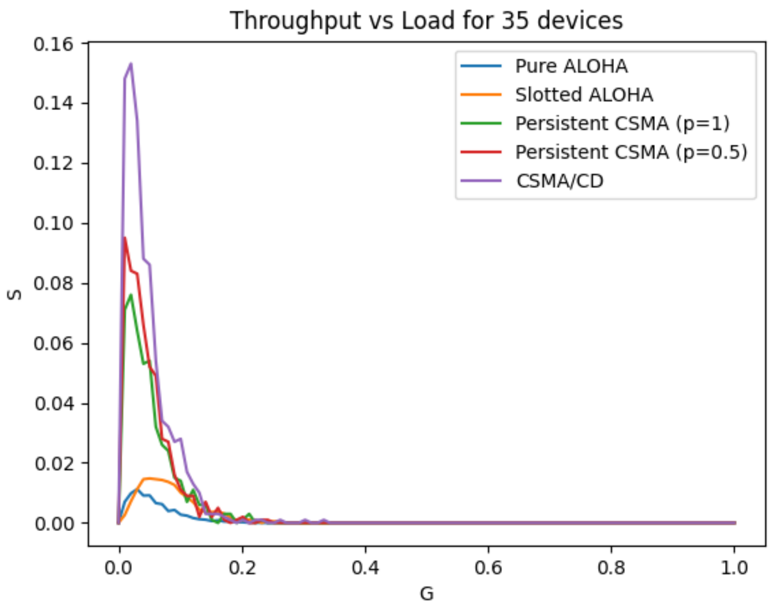

Figure 7: Simulation results for 35 devices

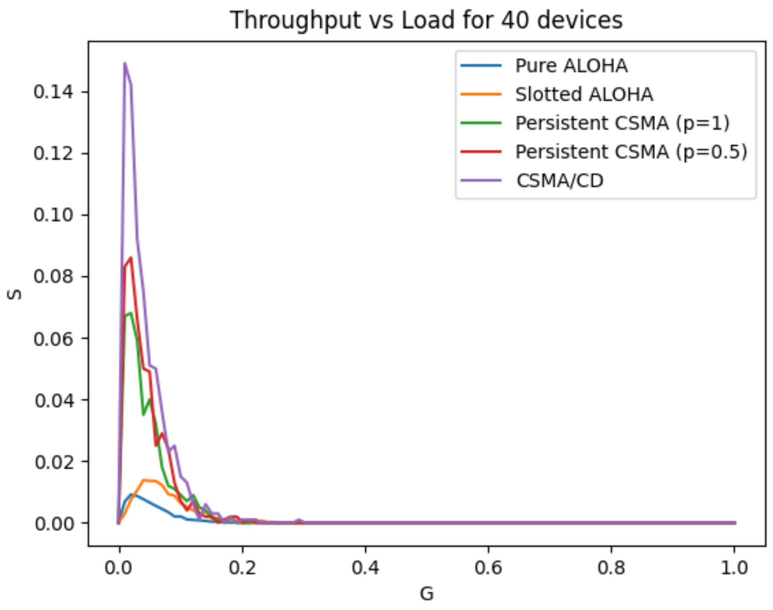

Figure 8: Simulation results for 40 devices

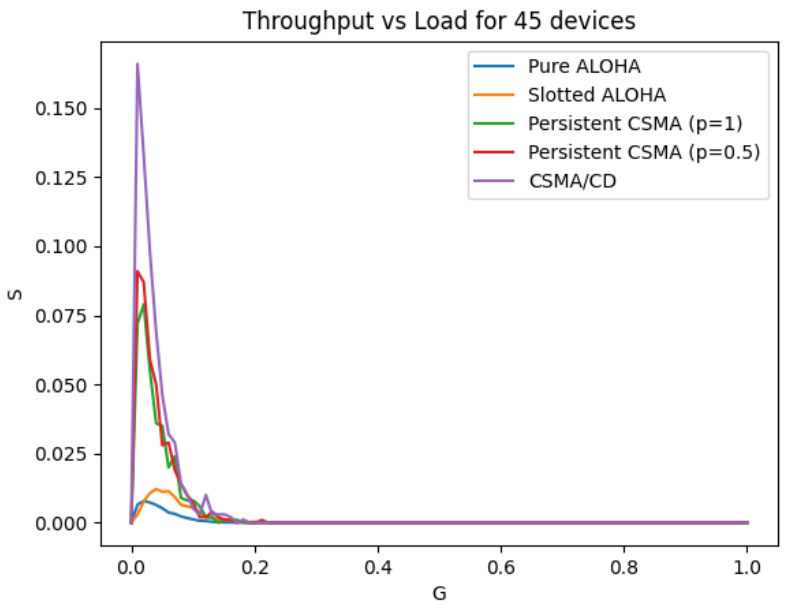

Figure 9: Simulation results for 45 devices

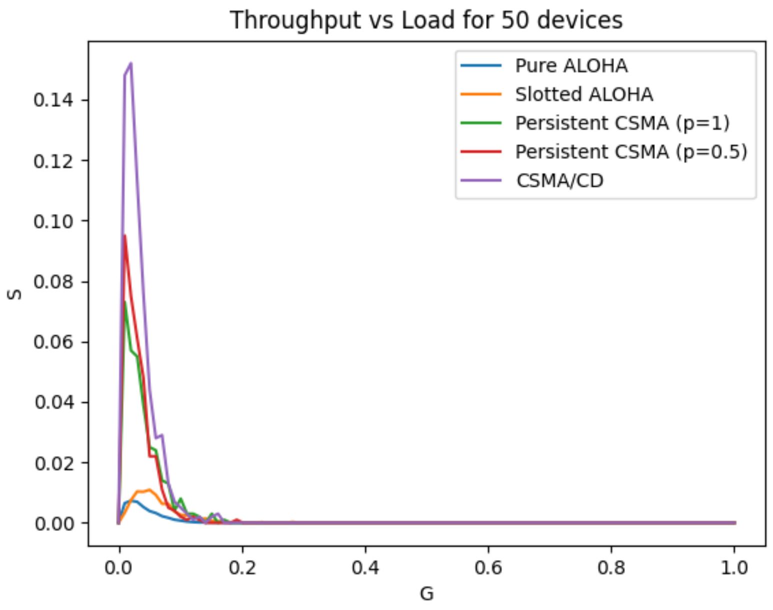

Figure 10: Simulation results for 50 devices

Problems with the Simulation

In the conducted simulation using the provided Python script, the focus was on evaluating the performance of multiple access protocols, including Pure Aloha, Slotted Aloha, Persistent CSMA, and CSMA/CD. Each protocol was individually simulated within the Simulation class, employing sophisticated mathematical calculations involving probability distributions like Poisson distributions to model packet generation and transmission. However, it's essential to acknowledge that this simulation is a mathematical model rather than a true representation of real-world scenarios. As such, it may exhibit inaccuracies, such as the unexpected behavior of CSMA protocol throughputs declining after the load parameter reaches 1, rather than approaching 1 as anticipated. Despite these limitations, the simulation results, saved as .png files for each protocol and device count ranging from 5 to 50 in increments of 5, offer valuable insights into protocol behaviors under various network conditions. This understanding can inform network optimization efforts and design decisions.

The expected results are given in the image below:

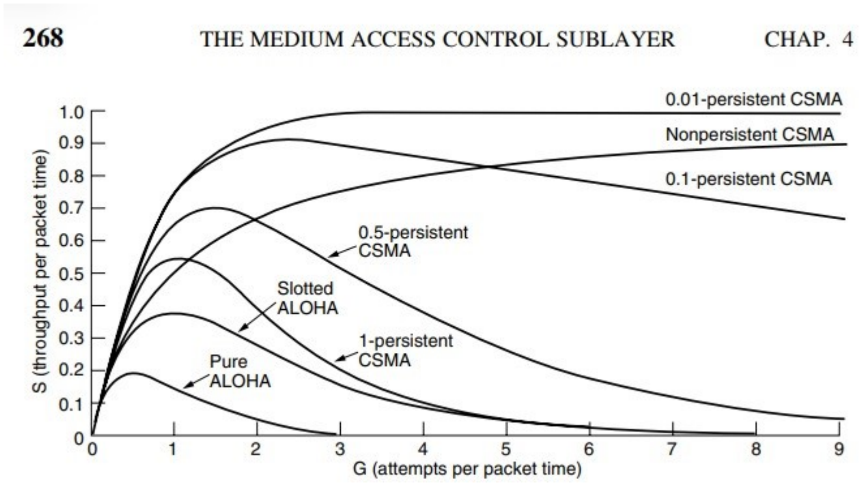

Figure 11: Expected results of the protocols