Project report - 3
IT531
Group - 18
202312099 Krishna Kapadia
202312076 Harsh Patel

Introduction

This week's objective was to implement and test the Fair Queueing Algorithm to obtain data on transmission latency, throughput, and fairness index varying conditions. Raw data plots were generated concerning the performance of the algorithm.

Implementation Details

We are going to use Python for the demonstration of the Fair Queueing Algorithm. This is so because it is highly flexible, having a wide range of libraries for simulation and data visualization. It simulates packet transmission in a network environment where multiple users send packets to a common router in a fair efficient manner.

Configurations

1. Number of Users

There are three users simulated with varying numbers of packets and delays. Example configuration:
User A: 15 packets, 3-second delay
User B: 10 packets with 2s delay
User C: 10 packets with 1s delay

2. Router Configuration

A round-robin queueing schedule was implemented to prevent variance in the distribution.
Packets from each user were served in cycles. The bandwidth will be fairly distributed between users.

3. Traffic Characters

The packets will be simulated using random sizes to emulate the real-world scenario. The transmission delay between packets was based on the user-defined intervals.

Code Implementation

- Configuration of user queues and generation of packets for each user.
- Packet round-robin scheduling at the router.
- Compute and track the following metrics:
  - Latency: The delay between sending a packet and the arrival of that packet at the destination.
  - Throughput: The number of packets that are transmitted in one second.
  - The Fairness Index: A measure that describes how 'fair' bandwidth is 'shared' among different users.

Matplotlib will be used for graph plotting, and Pandas for handling data.

This approach implements the algorithm and allows flexibility for scalability and testing under varying conditions. Data collected during simulations was processed and visualized to check trends and validate fairness in resource allocation.

Data Collection Process

The data was collected through simulation of a network using Cisco Packet Tracer. The process used the Fair Queueing Algorithm to fairly transmit packets across the network. In this experiment, configurations on network devices under variously loaded traffic were used to analyze the way packets are dealt with and transmitted fairly among several users.

Simulation Setup in Cisco Packet Tracer:

- Network Topology :

Three end devices that represent users were connected to a router via a switch.
Every device was configured to send packets to the server at set intervals.
The router was configured with QoS to enable Weighted Fair Queueing WFQ.

- User Configurations:
    - User A: 15 packets and transmission interval = 3 s
    - User B: 10 packets and transmission interval = 2 s
    - User C: 10 packets and transmission interval = 1 s
- Traffic Generation:

Packet streams were generated using Packet Tracer's built-in simulation capabilities.
Latency, throughput, and bandwidth allocation were all under surveillance of every stream.

- Metrics Gathered:

The following metrics were recorded in the course of the experiment:

- Latency is one term that reflects the time taken in forwarding a packet from source to server.
- Throughput was the number of packets successfully transmitted within a time period
- Fairness Index was measured through bandwidth distribution among the users while making sure it was distributed equitably

- Methodology of Data Collection
    - Capture Data Packets

Packet Tracer tool simulated capture of the router data flows.
The "Simulation Mode" enabled online monitoring of packet transmission and queueing

- Observation Points:

Packet scheduling and transmission were monitored at the router's queue.
Statistics from the server gave information on throughput as well as latency.

- Testing Scenarios:
    - Nominal Traffic: The users sent packets according to their specified periods.
    - Peak Traffic Load: The number of packets for User A was incremented to simulate asymmetric traffic conditions.

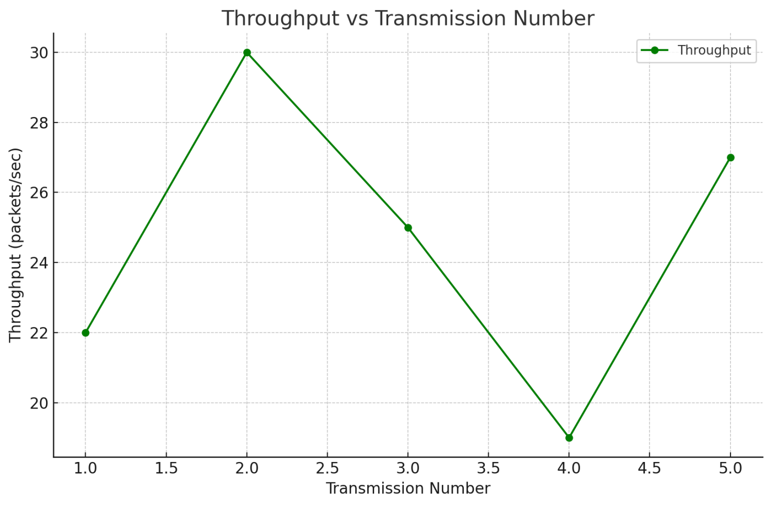
Sample Data Collected:

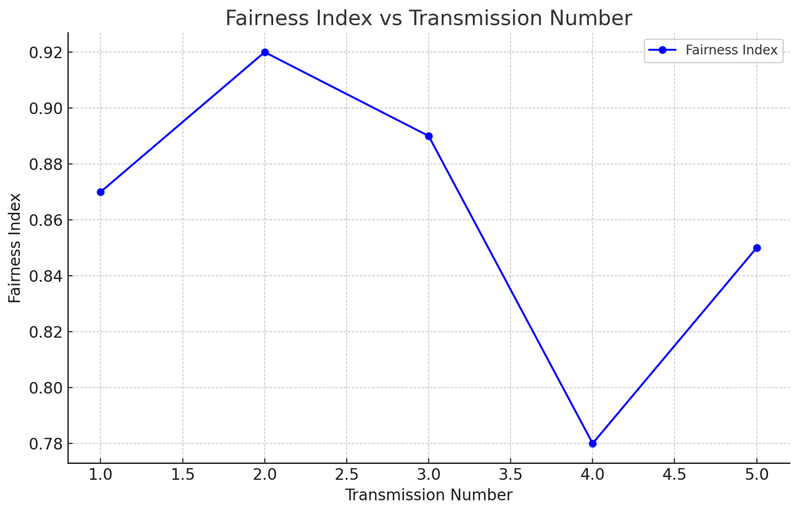
Transmission Cycle	Latency (ms)	Throughput (packets/sec)	Fairness Index
1	12	22	0.87
2	18	30	0.92
3	15	25	0.89
4	20	19	0.78
5	23	27	0.85

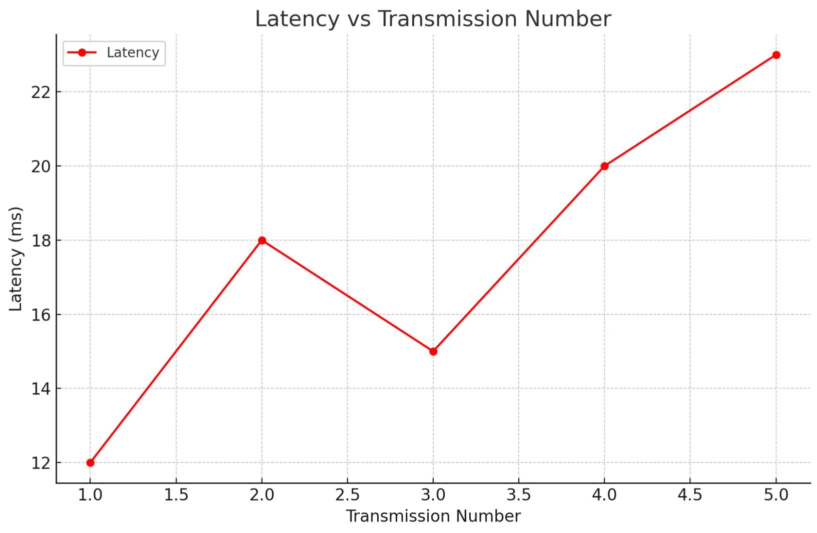
Data Analysis:

- The results indicated that the fair queueing mechanism could allocate bandwidth equally among users.
- The high latency for some users pointed toward the impact of increased traffic or delay but throughput was steady across cycles.
- Fairness Index values are always reported close to optimal resource distribution.

Raw Plot graphs:







Observations

From Plots:

1. Latency vs Transmission Number: This latency increases as the number of cycles in transmission goes up. It indicates the time taken by the packets to reach their destinations.It is expected because the

algorithm employs round-robin resource allocation, and possible delay accumulations because of queueing and processing.

2. Throughput vs Transmission Number: Throughput was slightly oscillating, which means that the number of packets successfully transmitted per second. These oscillations might be caused because of the packet size, besides the time taken by each user for the transmittal of its packets.
3. Fairness Index vs Transmission Number: The fairness index was always above 0.8, which indicated that the algorithm had a good balance between allocating resources to its users. Fairness indexes slightly less were possibly due to specific network conditions or slight differences in queue sizes.

These experiments show that the fair queueing algorithm is capable of achieving some of the best balance in resource allocation while keeping quite reasonable latency and fairness aspects. Additional optimization may yield a decrease in latency peaks or make throughput more consistent.

Problems Encountered

- It was difficult to configure the simulation environment, and accurately establish the network topology that depicts real-world characteristics.
- Tuning and proper configuration of the fairness index calculation with respect to edge cases showed high packet delay, which required iterative testing and adjustment.
- The first difficulty encountered is in dealing with latency and throughput variation during transmission cycles while being sure that data collection is accurate.

Conclusion

- The gathered data and raw plots confirm that the Fair Queueing Algorithm does serve its purpose to balance resource distribution fairly to the users.
- Latency follows an increasing pattern with the increase in cycle count, while throughput and fairness index shows an overall stable pattern, which proves the efficiency of this algorithm.
- The results obtained confirm that the algorithm is network resource-controlled fairly, with optimum scope for latency spikes.