Omnet Simulation Setup:

I utilized the provided FIFO queue sample project as my starting point. I needed to make the following changes to ensure I had a working simulation for the task:

1. **FifoNet.ned**

   This file defines the ”FifoNet” setup. I created three identical queues consisting of a source node, a FIFO server, and a sink node, in which all traffic flows through the three nodes; this setup is shown in Figure #1.

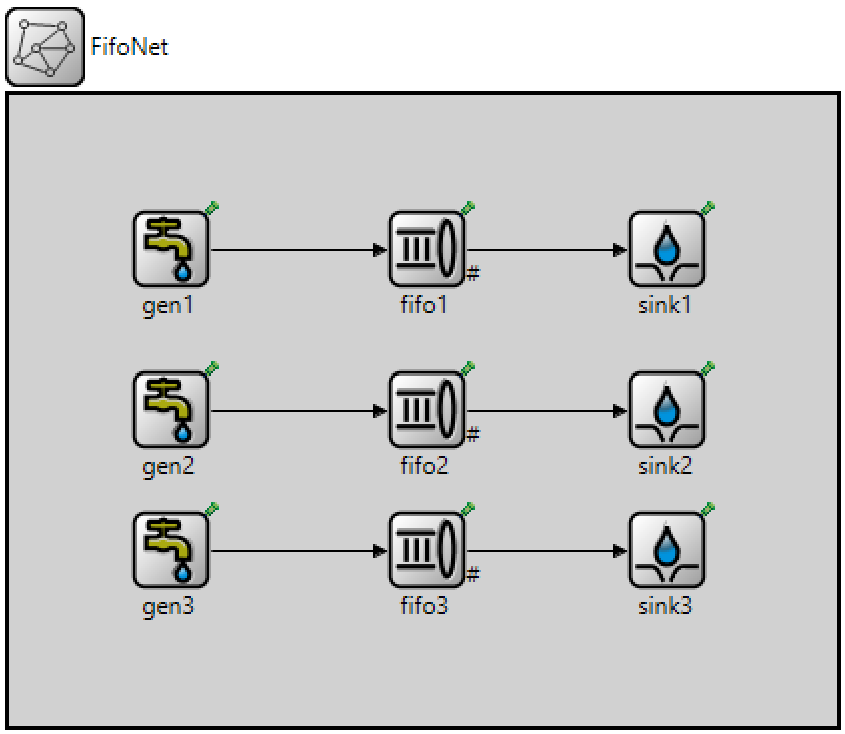

Figure 1: Diagram showing the three queue setup.

   When a packet arrives at the sink, it generates a packet delay data point as defined in Eq. #1

$$Packet\,Delay = time_{arrival} - time_{created} \tag{1}$$

2. **omnetpp.ini**

   The modifications to this file specified each queue's individual parameters. Each queue had a **service time** of $t_s = 0.75\,seconds$. I differentiated the arrival rates of each queue by specifying the interarrival time from each generator/source. These times are shown below:

Queue #:	Interarrival Time (seconds)
1	1.0
2	0.50
3	0.25

Table 1: Interarrival Times

   The final change was to set the **sim-time-limit** to one hour.

Results & Analysis:

Utilization:

Given the interarrival times and service times of each queue, I calculated values for λ, μ, and ρ for each queue using the below relationships:

$$\lambda = \frac{1}{Interarrival\,Time} \qquad \mu = \frac{1}{Service\,Time} \qquad \rho = \frac{\lambda}{\mu}$$

Note, each queue has a service time of 0.75 seconds, and the interarrival times stated in Table #1

Queue #:	λ	μ	ρ
1	1.00	$1.\overline{33}$	0.75
2	2.00	$1.\overline{33}$	1.50
3	4.00	$1.\overline{33}$	3.00

Table 2: Utilization

From these results, I would expect Queue #1 to have a negligible delay. Because the equation used to calculate ρ breaks down with $\rho > 1$, the calculated utilization of Queues #2 and #3 is unattainable. I would expect Queues #2 and #3 to be fully utilized, with $\rho_{effective} = 1$.[1] However, Queues #2 and #3 will have a queue length which will grow to infinity. This makes sense given the meaning of each variable: for Queue #1 - packets arrive slower than they are serviced, for the other two queues - packets arrive faster than they can be serviced.

System Delay

Figure #2 shows the system delay of each queue.

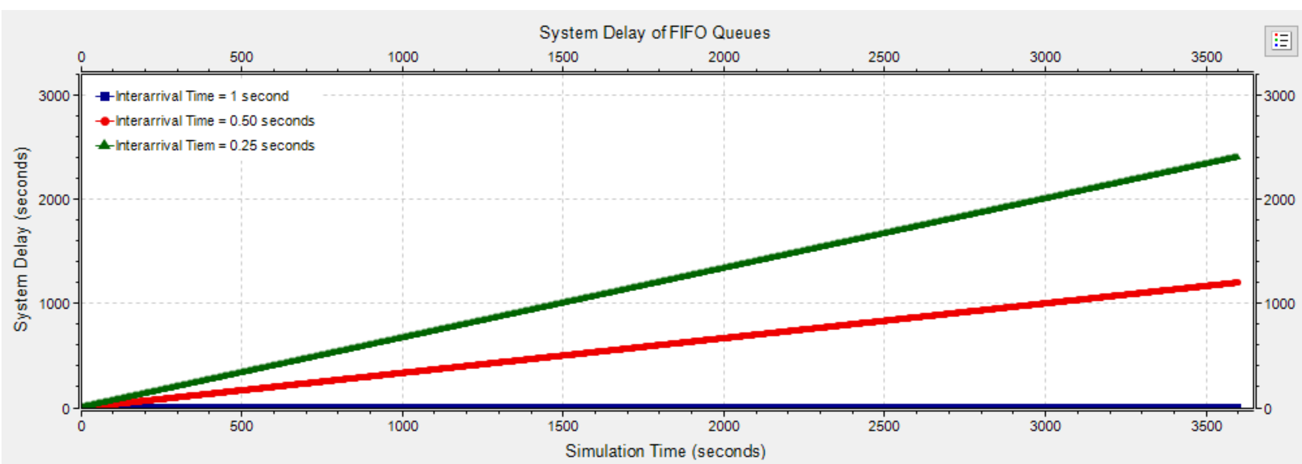

Figure 2: System Delay of Queues #1-3

Packets in Queue #1 experienced $delay = service\,time$. However, for Queue #2 and Queue #3, their system delays increased rapidly as the queue length increased. Despite the differences in arrival rates between Queue #2 and #3, they both serviced 4800 packets. Queue #1 serviced each packet which arrived, servicing a total of 3600 packets. One can see the relationship between the arrival rates, service rates, and throughput. When each packet that enters the system is serviced, as in Queue #1, $throughput = \lambda$; when the queue grows infinitely, as in Queues #2 and #3, $throughput = \mu$.

[1] $\rho_{effective}$ is the actual utilization of the server, calculated as $1 - p_{idle}|_{p_{idle}=0} = 1$

Queue Length & Queue Time

Tables #3 and #4 below show the raw data for the queue length and queue time for each queue.[2]

Queue #	Count	Mean	Std. Dev
1	1	0.0	0.00
2	12001	1200.0	692.91
3	19201	4800.3	2771.50

Table 3: Queue Length

Queue #	Count	Mean	Std. Dev
1	3601	0.0	0.00
2	4801	600.0	346.52
3	4801	1200.0	693.04

Table 4: Queue Time

The data shown above is indicative of the system delays experienced by each packet. When the server can keep up with/stay ahead of the queue, arriving packets have no queuing delay: there is no queue! However, once the queue forms, if the arrival rate remains faster than the service rate, the queue simply continues to grow, causing increasing system delays for each packet.

Conclusions

This project demonstrated the effects of using deterministic arrival times and service times, which leads to deterministic arrival and service rates. As long as the service rate is faster than the arrival rate, the system will never queue. However, with arrival rates faster than the service rate, the system will become unstable, with the queue growing indefinitely. The queue lengths of the unstable systems grow with respect to their arrival rates, with a corresponding increase in system delay as the queue length grows. For any systems which always maintain a queue (such as #s 2 and 3), their servers will be fully utilized. Thus, for a system with deterministic rates, the optimal configuration would be with $\mu = \lambda \rightarrow \rho = 1$, which maintains full utilization without a queue.

[2] The count seems increased by 1, but is caused by the packets arriving at the end of the simulation, that had not been serviced.

Appendix A: Edited Files

```
// FifoNet.ned:
network FifoNet
{
    submodules:
        gen1: Source {
            parameters:
                @display("p=81,77");
        }
        gen2: Source {
            parameters:
                @display("p=81,157");
        }
        gen3: Source {
            parameters:
                @display("p=81,227");
        }
        fifo1: Fifo {
            parameters:
                @display("p=209,77");
        }
        fifo2: Fifo {
            parameters:
                @display("p=209,157");
        }
        fifo3: Fifo {
            parameters:
                @display("p=209,227");
        }
        sink1: Sink {
            parameters:
                @display("p=329,77");
        }
        sink2: Sink {
            parameters:
                @display("p=329,157");
        }
        sink3: Sink {
            parameters:
                @display("p=329,227");
        }
    connections:
        gen1.out --> fifo1.in;
        fifo1.out --> sink1.in;

        gen2.out --> fifo2.in;
        fifo2.out --> sink2.in;

        gen3.out --> fifo3.in;
        fifo3.out --> sink3.in;
}
```

Defining the network in FifoNet.ned

```
// omnetpp.ini
[General]
description = "3 Seperate Arrival times, same service times"
network = FifoNet
sim-time-limit = 1h
cpu-time-limit = 300s
#debug-on-errors = true
#record-eventlog = true
**.gen1.sendIaTime = 1s
**.fifo1.serviceTime = 0.75s

**.gen2.sendIaTime = 0.50s
**.fifo2.serviceTime = 0.75s

**.gen3.sendIaTime = 0.25s
**.fifo3.serviceTime = 0.75s
```

Omnetpp.ini edited to run the three queues with separate parameters