Programming Assignment 1

CS544: Topic in Networks

22nd Feb 2024

<u>Comparing Different Scheduling Algorithms for Packet transmission through a crossbar switch</u>

210101074 – Parth Kasture

210101078 – Pranav Jain

Problem Statement

We have a crossbar switch with N input ports and N output ports. In every time slot, packets are generated at every input port with some generation probability and then the generated packet follows respective scheduling algorithms, and they are forwarded accordingly to an output port. Packets are buffered at input and output ports.

Assumptions

We have made following assumptions

- Each time slot is 1 sec long and transmission of each packet takes 1 time slot.
- Packet generation probability is derived from uniform distribution, and a packet is generated only if the probability is less than a threshold value.
- A random output port is selected for each packet following uniform distribution.
- For delay calculations, we have considered that packets are generated at the start of a time slot, the offset has been ignored.

Traffic generation

The randomness in the output port and start time in generated traffic follow uniform distribution.

```
uniform_real_distribution<double> timeOffsetDistribution(0.001, 0.01);
uniform_real_distribution<double> probToTransmit(0.0, 1.0);
uniform_int_distribution<int> destinationPortDistribution(0, N - 1);
```

```
rand_prob = probToTransmit(gen);
if (rand_prob <= packetgenprob)
{
    startTime = timeOffsetDistribution(gen);
    destinationPort = destinationPortDistribution(gen2);

    (*p) = {port, destinationPort, arrivalTime, arrivalTime + startTime};
    p->printPacket();
    input_queue[port].push_back((*p));
}
```

The above snippets are from the code for iSLIP algorithm.

Scheduling Algorithms

1. **INQ -** In this algorithm, if two or more packets are destined to one output port, we randomly select one packet for the output port and rest packets remain in the input buffer.

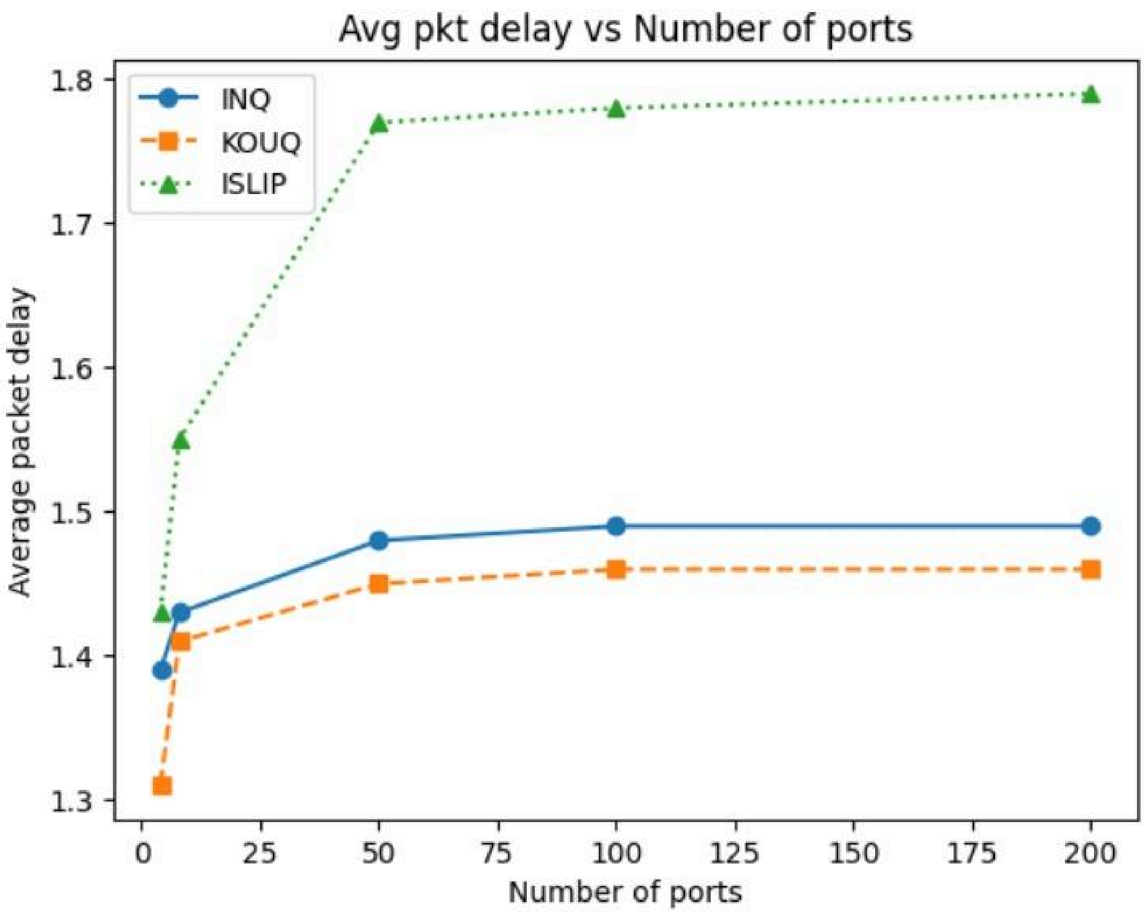
2. **KOUQ-** In this algorithm, for a time slot, out of all packets generated for an output port, only K random packets are buffered at the output port and the rest of the packets are dropped.

3. **ISLIP-** In this algorithm, an output port, upon receiving requests, selects the next one in a fixed round-robin schedule, beginning from the highest priority element. Following this selection, the output notifies each corresponding input about the grant status of their request. The pointer, denoted as gi, pointing to the highest priority element in the round-robin schedule, is incremented (using modulo N arithmetic) to the next position beyond the granted input only if the grant is accepted in a specific step.

Analysis

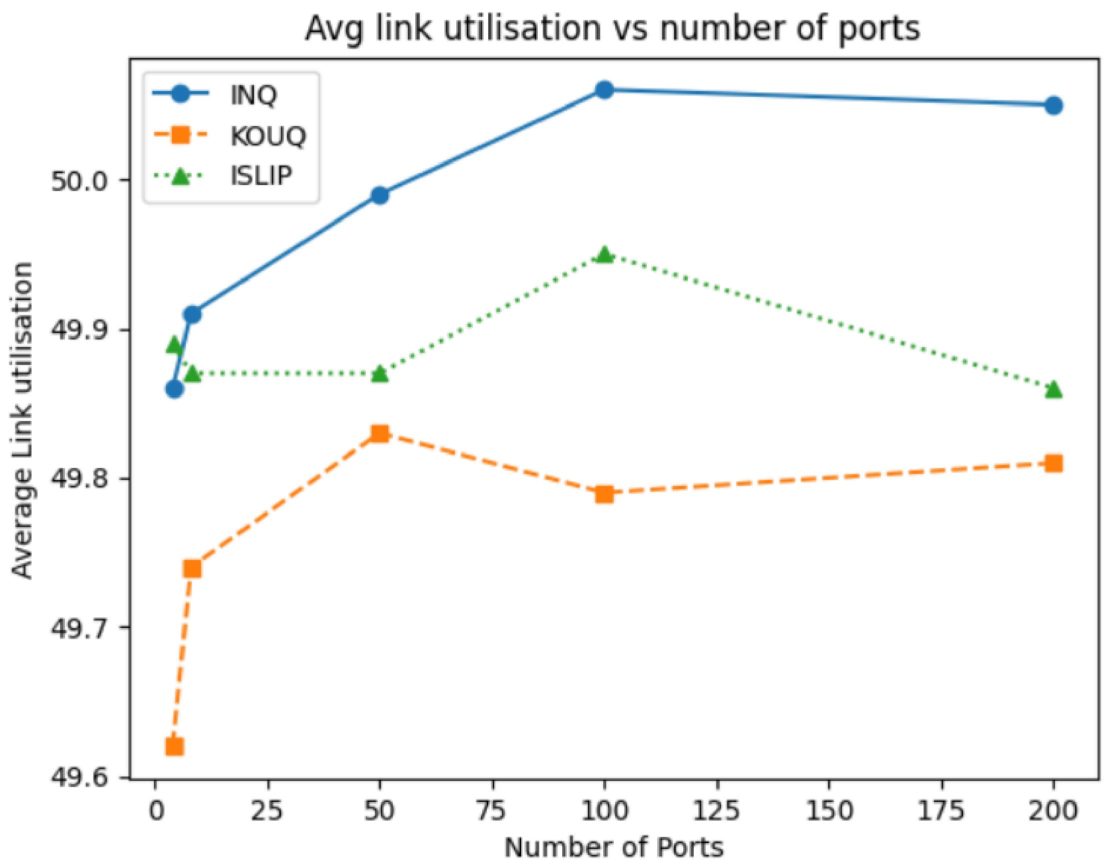
1. Avg packet delay vs Number of ports

First we varied the number of ports (N) with other values as default like generation probabilities = 0.5 and buffer size = 4. We observed that average packet delay is highest for ISLIP Algorithm, followed by INQ algorithm and lowest for KOUQ algorithm. Although average packet delay is almost the same for INQ and KOUQ.



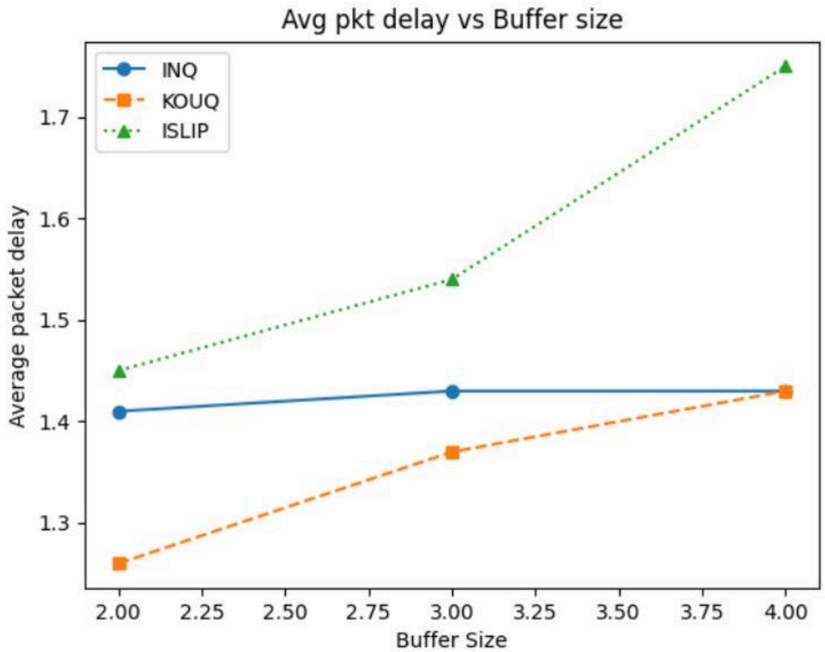
2. Avg link utilization vs Number of ports

Then, we measured the change in link utilization as we vary the number of ports (N) with other values as default, like generation probabilities = 0.5 and buffer size = 4. We observed that average link utilization is highest for INQ algorithm, followed by iSLIP algorithm and lowest for KOUQ algorithm. The link utilization is saturated for large number of input ports.



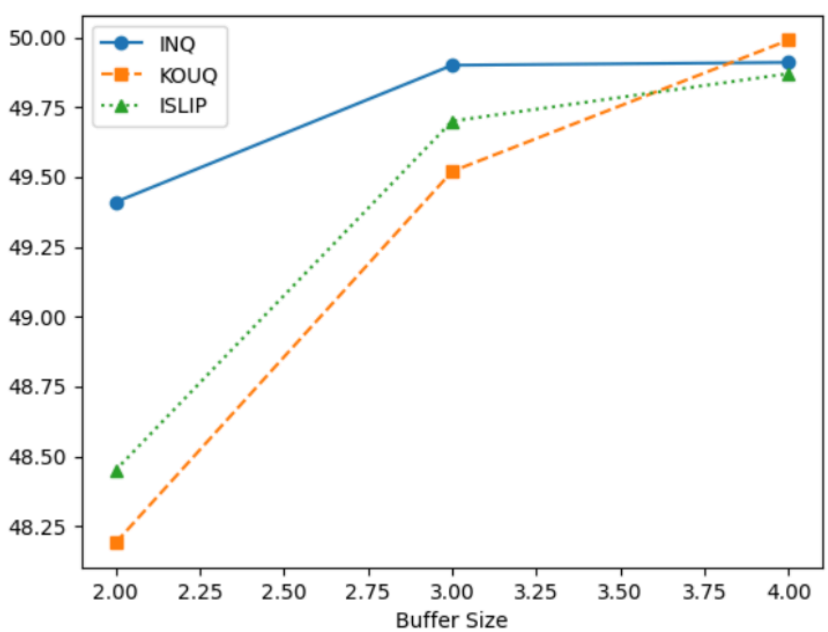
3. Average packet delay vs Buffer size

Then we kept N = 8 and other default values and varied buffer size of input and output queues. We observed a slight increase in average packet delay in all algorithms and link utilization was almost again similar. This increase in delay can be attributed to the fact that some packets will have longer queuing delays due to increased buffer size. The observations are plotted in the following graph.



4. Average link utilization vs Buffer size

For N = 8 and other default values, we varied buffer size of input and output queues. We observed a slight increase in average linked utilization in all algorithms. On increasing the buffer size the link utilization increases because the packets in the buffer makes the output port busy. The observations are plotted in the following graph.



5. Drop probability vs K

We varied the K factor for the KOUQ algorithm and observed the K drop probability which is the probability per slot that more than K packets were generated for an output port. We observed that for very less K factor a lot of packets are dropped because of less K value. As K increases drop probability decreases, and almost reduces to 0 for large K.