Simulation of the performance of a two-queue system

Sanjay Vasudevan Sunderrajan, Avinash Nagarajan

Abstract: ***In this simulation we are going to simulate a two queue system to measure the performance metrics that is, blocking probability of incoming packets, the average queue length and the average waiting time of a packet inside the queue. The performance metrics would be measured with the parameters; arrival rate (λ), service rate (μ), traffic load (ρ). This project focuses on simulation of packets through two queues with FIFO policy and each queue has its own condition to accept the incoming packets. The first queue accepts packets randomly and uniformly distributed rate. The second queue accepts the packet when the queue has minimum number of packets in it. We're going to be plotting graphs based on simulation and theoretical description of the problem and hope to achieve a graph overlapping one another to justify that the simulation has gone right. We would be seeding random packets and feeding it to the queue and the server would service the incoming packets with a service rate and the arrival of packets has a specific arrival rate. We are here to calculate the ratio of the dropped packet vs the total packets sent to the queue as well. We would be performing the sensitivity analysis through packet level simulation and identify the key parameters which affects the performance metrics.***

I. Introduction

In this simulation, we simulate the performance of a two queue system as shown in Figure 1, under various packet-assignment strategies. There are two queues as shown in the figure. Each queue has a capacity of 10 packets (including the one in the server). A packet assignment strategy involves the thinking of how a packet generated would be sent to a particular queue.

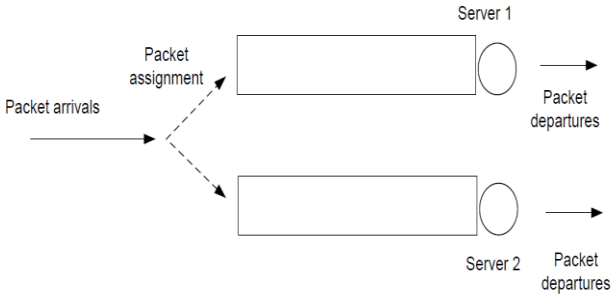

Figure 1: Two queue switch

There are two specific strategies involved in the selection of packets. The first one involves the generation of packets with random selection, which assigns the incoming packet uniformly randomly to one of the queues. Another packet assignment strategy is minimum-queue strategy, which assigns the packet to the queue of the minimum length.

II. DESIGN

We design the simulation in such a way that it follows the protocols of the packet assignment. It has two queues working and each queue is of the model M/M/1. The arrival rate follows poison process and the service rate is the time taken by the server of a queue to serve one packet. There isn't any packet switching in between queues to swap the packets from queue 1 to queue 2 in case queue 2 has lesser packets when compared to queue 1. We plan on to construct individually all the components of the simulation. The components may include, random exponential value generation. Then we would have a function to decide upon the packet assignment, assign the packets to the rear of the queue and post it to the server to process it. The server serves the packet in the service time specified and it would accept the next packet. The parameters involved are the arrival rate of the packets and the service rate of the packets. The calculations that we are supposed to simulate is the blocking probability where the average waiting time, average queue length and blocking probability which is the ratio of the totally dropped packets on both queues by the total number of packets in the queue.

III. SENSITIVITY ANALYSIS

A. Performance metric (Blocking Probability) vs Performance parameter Arrival rate (λ)

The graph below depicts the comparison with the X axis the range of the parameter and the Y axis with the performance metrics.

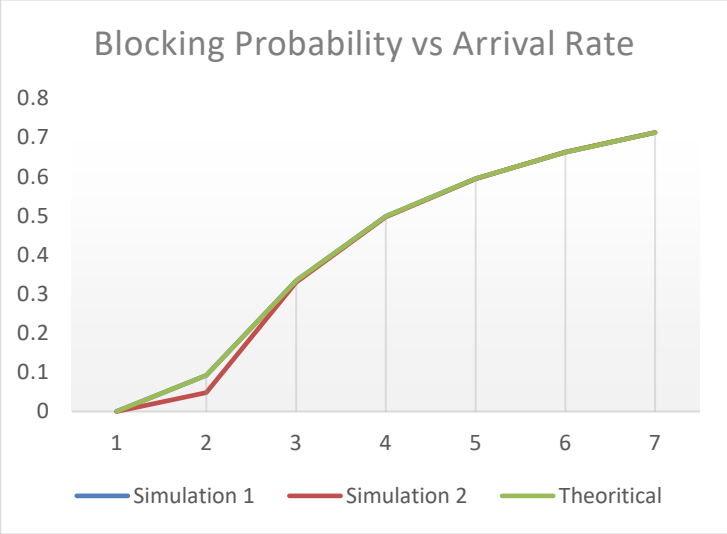

Figure 2 : Blocking Probability vs Arrival Rate

From this figure we can see that the Simulation 1 and the Theoretical comparison for the random queue strategy follow almost similar values. Considering the values having being rounded off because of the calculation power we can see the difference between the simulation 1 and simulation 2 anyway. The min-queue strategy and random assignment strategy at the start; min-queue is more efficient but as the arrival rate increases , it differs by very minute seconds as that is significant on the point of view of a computer and network performance.

B. Performance metric (Blocking Probability) vs Performance parameter Service rate (μ)

The graph below depicts the comparison with the X axis the range of the parameter and the Y axis with the performance metrics.

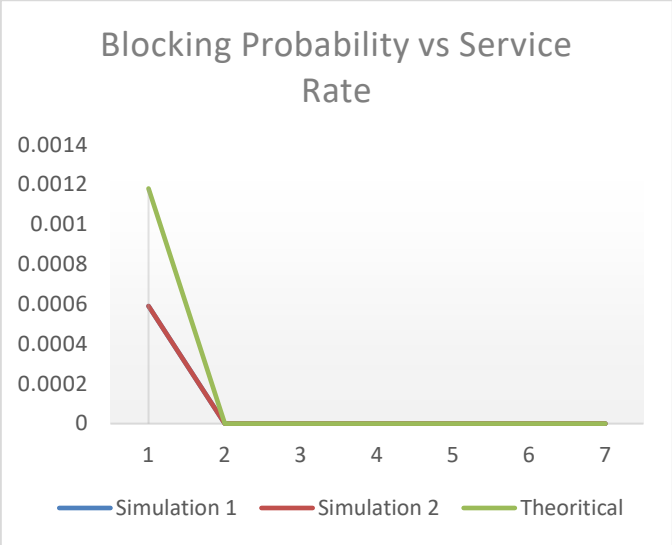

Figure 3: Blocking Probability vs Service Rate

This shows that the blocking probability drops to 0 if the service rate is much higher than the arrival rate.

C. Performance metric (Blocking Probability) vs Performance parameter Traffic Load (ρ)

The graph depicts the comparison with the X axis the range of the parameter and the Y axis with the performance metrics.
We can notice that the blocking probability is higher as the arrival rate or the traffic load is increased. The traffic load is calculated by the arrival rate over 2 times the service rate for a component.

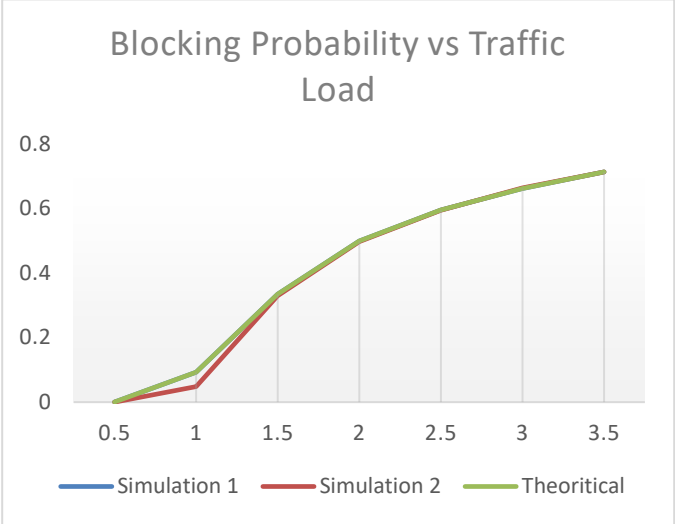

Figure 4: Blocking Probability vs Traffic Load

D. Performance metric (Average Queue Length) vs Performance parameter Arrival rate (λ)

The graph below depicts the comparison with the X axis the range of the parameter and the Y axis with the performance metrics.

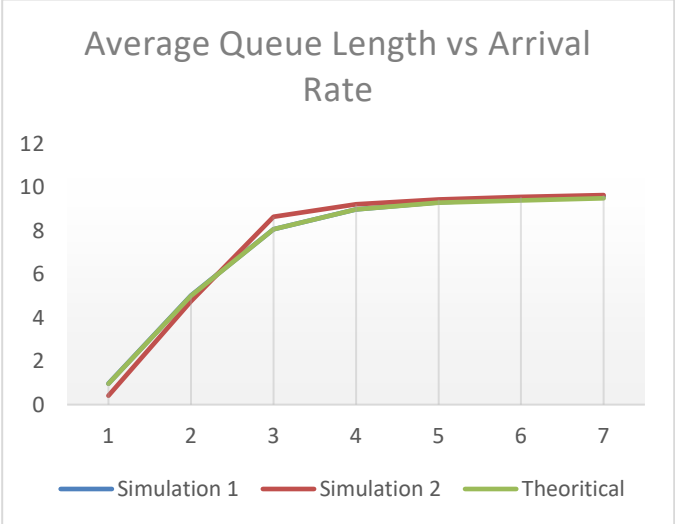

Figure 5: Average Queue Length vs Arrival Rate

We can notice that the average queue length increases as the rate of arrival increases and it goes by a stagnant average of 9 as the capacity of the queue is only 10.

E. Performance metric (Average Queue Length) vs Performance parameter Service rate (μ)

The graph below depicts the comparison with the X axis the range of the parameter and the Y axis with the performance metrics.

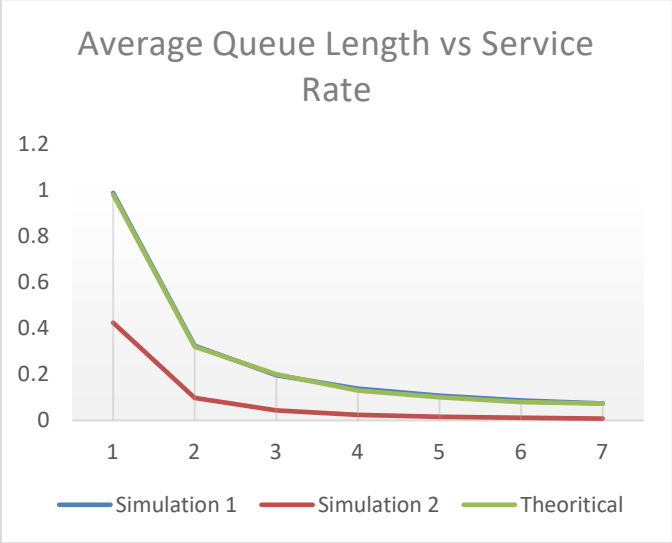

Figure 6: Average Queue Length Vs Service Rate

There is a huge difference in both the simulation where the min-queue strategy has much better performance handling the packets than the random queue assignment.

F. Performance metric (Average Queue Length) vs Performance parameter Traffic Load (ρ)

The graph below depicts the comparison with the X axis the range of the parameter and the Y axis with the performance metrics.

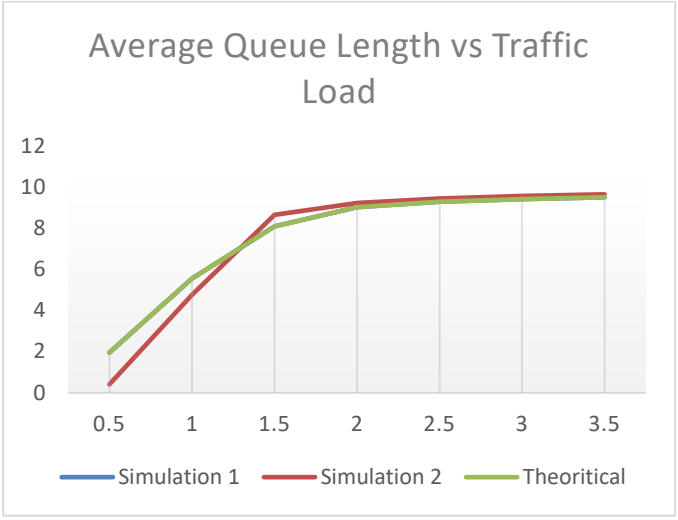

Figure 7: Average Queue Length vs Traffic Load

We can notice that the average queue length increases as the traffic load increases as well.

G. Performance metric (Average Waiting Time) vs Performance parameter Arrival rate (λ)

The graph below depicts the comparison with the X axis the range of the parameter and the Y axis with the performance metrics.

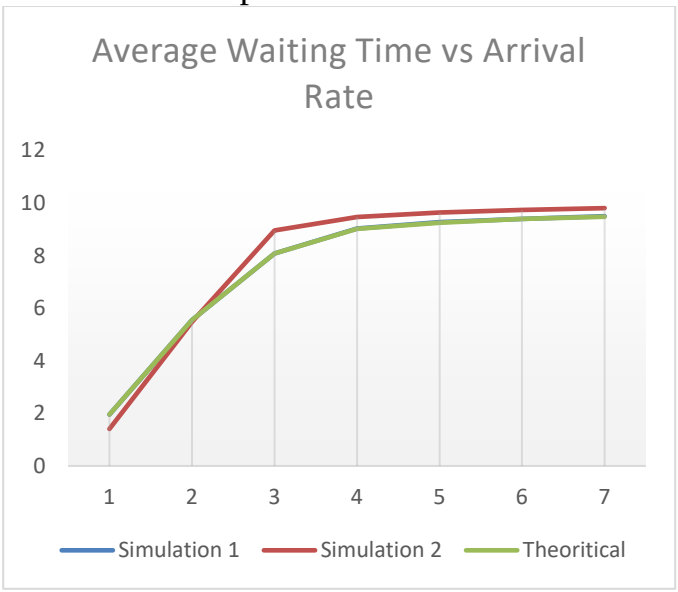

Figure 8: Average Waiting Time vs Arrival rate

We can see that the average waiting time for a packet is higher as the arrival rate increases.

H. Performance metric (Average Waiting Time) vs Performance parameter Service rate (μ)

The graph below depicts the comparison with the X axis the range of the parameter and the Y axis with the performance metrics.

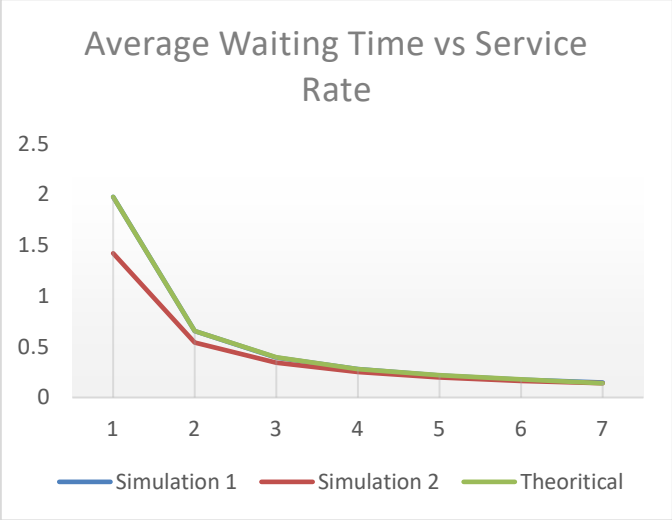

Figure 9: Average Waiting Time vs Service Rate

We can see from the graph that the min-queue does better when the service rate is much higher.

I. Performance metric (Average Waiting Time) vs Performance parameter Traffic Load (ρ)

The graph below depicts the comparison with the X axis the range of the parameter and the Y axis with the performance metrics.

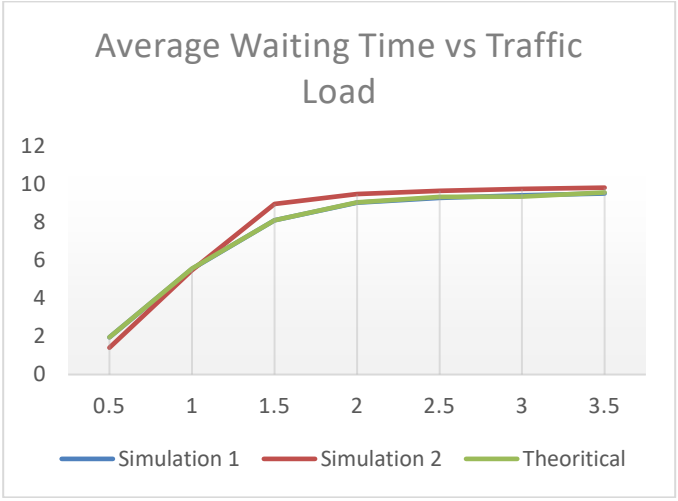

Figure 10: Average Waiting Time vs Traffic Load

We can notice that the average waiting time in a queue is high when the traffic load is high.

IV. CONCLUSION AND INSTRUCTIONS FOR EXECUTION

We can conclude that the sensitivity analysis done for the random queue strategy and the min-queue strategy is right because the theoretical calculation for the queue's for the random queue assignment co-relates with the simulated result for each and every performance metric. The important factor for a network with a system like this to perform better is when the service rate is higher than that of the arrival rate. If the significant difference between the service rate and the arrival rate is present, where the service rate being the higher value we can notice that the system performs much better. The important factor for performance on certain scales are also the queue to which the packet is sent to.

The min-queue strategy performs slightly better than the random queue strategy.

The program can be executed by running random.c and minQueue.c using gcc compiler. It also requires –lm to be added at the end to make use of the linker to link the program to the math.h library.
Example : gcc –o random random.c –lm.

Then the requirement is to enter the service rate and the arrival rate for the particular queue assignment strategy and the program will generate 10 independent seeds calculating the average waiting time, average queue length and the blocking probability.