**DISCRETE TIME QUEUE**

**Jaya Godugu**

SBU ID#108209853

Computer Communication Networks

**Project Description**

This project produces a discrete time simulation of a queue as shown in figure. Time is slotted on the input and the output. Each input packet follows a Bernoulli process. In a given time slot the independent probability that a packet arrives in a time slot is *p*, while the probability that the packet will be serviced is *q*. One packet fills one time slot.

*p*

*q*

* The queue can store up to four packets.
* Packets are processed on a first come – first served basis (FIFO).
* When a packet is serviced all other packets in a queue (if any) are shifted

instantaneously towards the output.

Each slot departures from the queue are processed before arrivals.

The following outputs were collected from the program:

1. Average throughput
2. Averaged delay (for values of p from p=0.02 to p=1, for a fixed value of q=0.75)
3. Comparison of Average throughput V/s values of p & Average delay V/s values of p

Start

Implementation:

1. Flowchart:

Initialize & declare variables & Buffer, Delay Arrays

For p=0.02; p<1; p+=0.02

1

{For 10,000 slots}

Generate random number ‘r’

Add the delay value of the departing packet to D\_count

Shift elements of the Buffer[i] ,FIFO

Shift the delay queue T\_Delay[i]

Increment T\_Count

Yes

Is r <0.75 i.e. is the packet in the queue to service?

No

Generate random number ‘r’

Place the new packet in the queue i.e. Buffer[i]

Set the delay for this packet to 0

Yes

Is r < p i.e. has a packet arrived ?

No

Increment the delay array T\_Delay[i] for each i / packet

No

Is N = 10000?

Yes

Average Throughput = T\_Count/10000

Average Delay= D\_Count/T\_Count

No

Is p = 1?

1

Yes

Stop

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**Program:**

* Language used: C Programming
* Packages: Standard libraries

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdio.h>

#include <stdlib.h>

main()

{

int slot,i,j,h;

float Delay,Throughput,r,n=1000,D\_Count=0,T\_Count=0,p,T\_Delay[4];

int Buffer[4];

FILE \*ftpt, \*fdel;

if ((ftpt = fopen("t\_file", "w+")) == NULL){

printf("can't open t\_file \n\r");

exit(0);

}

if ((fdel = fopen("d\_file", "w+")) == NULL){

printf("can't open d\_file \n\r");

exit(0);

}

for(i=0;i<4;i++){

T\_Delay[i] = 0;

Buffer[i] = 0;

//printf("\n the delay is %f for %d cycle",T\_Delay[i],i);

}

for (p=0.02;p<1;p+=0.02){

for(slot=0; slot<10000; slot++){

r=((rand()%1000)/n);

//printf("\n The random no is %f",r);

/\*Checking if the packet has to depart from the buffer \*/

if (r<=0.75 && (Buffer[3] != 0)){

D\_Count = T\_Delay[3] + D\_Count;

for(j=2; j>=0;j--){

Buffer[j +1] = Buffer[j]; /\*shifting elements in the queue \*/

T\_Delay[j+1] = T\_Delay[j];}

Buffer[0] = 0;

//T\_Delay[3] = 0;

T\_Count = T\_Count + 1;

//printf("\n the value of the Throughput counter is %f",T\_Count);

}

r = ((rand()%1000)/n); /\* Checking for arrival \*/

//printf("\n The random no is %f",r);

/\*checking for an arrival \*/

if(r<=p){

/\*FIFO Concept \*/

for (i=3;i>=0;i--){

if(Buffer[i]== 0){ /\* No Packet = '0' \*/

Buffer[i]= 1;

T\_Delay[i] = 0;

goto l1 ;

}

}

}

/\* for(i=0;i<4;i++){

printf("\n The elements after arrival are:eli[%d] %d \n", i, Buffer[i]);

} \*/

l1: for(j=0;j<4;j++){

if(Buffer[j] != 0){

T\_Delay[j] = T\_Delay[j]+1; /\*Assigining Delay \*/

//printf("\n The delay array values are %f ",T\_Delay[j]);

}

}

/\* for(i=0;i<4;i++)

{

printf("\n The delay array values are %f ",T\_Delay[i]);

} \*/

// sleep(1);

} /\* end slots \*/

printf("\n The T\_Count value is %f",T\_Count);

Throughput=(T\_Count/10000);

Delay= (D\_Count/T\_Count);

printf("\n The Throughput is %0.9f \n",Throughput);

printf("\n The Delay is %2.9f \n",Delay);

fprintf(ftpt,"%0.9f\n", Throughput);

fprintf(fdel,"%2.9f\n", Delay);

T\_Count = D\_Count = 0;

} /\* end p \*/

fclose(ftpt);

fclose(fdel);

} /\* end main \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**Program Description:**

This program emulates the discrete time simulation of a queue. With the independent probability that the packet arrives i.e. p & the service completion rate q one can determine the average throughput & delay for the give queue.

1. Clear the delay & Buffer Arrays.
2. For p=0.02 till 1
3. For slot =1 till 10000
4. Check for departure of the packet.
5. If a packet departs, add its delay value to a variable (i.e. cumulative sum) shift the input buffer queue & the relevant Delay array. If not go to step 6
6. Check for arrival
7. If there is an arrival designate the immediate available slot in the queue & assign a delay of 0.Else go to step 8
8. Increment the delay counter wherever packets are present in the buffer queue
9. Check if 10000 slots for one run of p have been satisfied
10. If yes calculate the average delay using the formulae. Else go to step 3

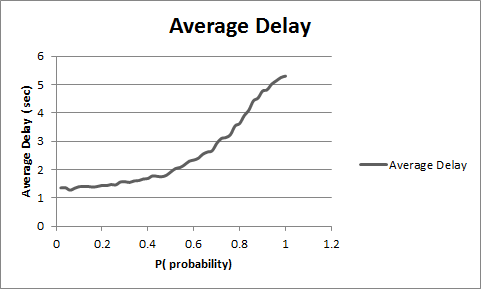
Average Throughput = Total no of packets serviced/ No. of time slots

Average Delay = Total # time slots spent in the queue/ Total # packets serviced.

1. Repeat until p=1

Graphs:

(a)



(b)

**Conclusion:** It is seen form the above graphs that as when the independent probability of arrival of a packet i.e. p increases the throughput. However as the maximum throughput being set in this case the Graph 1i.e. Throughput saturates at 0.75. Similarly as p increases the value of delay increases. The worst case of delay could be all the four packets being queued up and no serviced which in that case (4\*1/75 =16/3 sec) is the highest delay a packet could experience which is shown in Graph (b).