Simulation of the Iterative Probabilistic Scheduling with Parallel Processing

Avtar Singh, Dr. Sandeep Kumar
Research Scholar, Singhania University Pacheri badi,Jhunjhunu (Rajasthan)
Shree Atam Vallabh Jain College, Ludhiana

Abstract

The implementation of VOQ and Iterative Probabilistic Scheduling with Parallel Processing (IPSWPP) in ns-2 is setup as described. The simulator is configure to model a heterogeneous router processor system having the dispatching processor, multiple worker processor and one transmitting processors. To examine the performance of IPSWPP for processing variable length packets in a realistic system, we conduct the simulation by varying the mean packet size of the incoming packets.

Keywords

CBR; VBR; CBQ and MQAS.

1. Introduction

The network simulator 2 version 2.31 (ns-2.31) systems [15] was used to produce the simulations. Ns-2 (Network Simulator Version 2) is an object-oriented discrete event-driven simulator that can simulate a variety of IP networks. It implements network protocols such as TCP and UDP, traffic source behavior such as FTP, telnet, Web, CBR and VBR, router queue management mechanism such as RED and CBQ, routing algorithms like Dijkstra and more. The components of ns-2 are separated into two languages spaces: C++ and OTcl (Tcl script language with object-oriented extensions). Components implemented in C++ space run fast and can efficiently interpret packet headers and implement algorithms running over large data sets. Scripts in OTcl space run much slower but are flexible in changing, making them ideal for configurations, topology setup and manipulations of components in C++ space.

Because of the way ns-2 is structured, it was decided that the best way to design a link is to associate a queue with it directly. Hence, when a link is created between two nodes the buffer controlling access to the link is part of the link object in ns, rather than the node object, where it is located in the physical world. Therefore a VOQ can be installed or associated with a link to a node. This can be viewed as being equivalent to installing a VOQ at the input interface to a node. The way that this is done logically is shown in Figure 1.

The VOQ is installed before the node because of the way the node is implemented, all the traffic enters the node at the same point, once the traffic has entered the node, the link it arrived on is difficult to obtain.

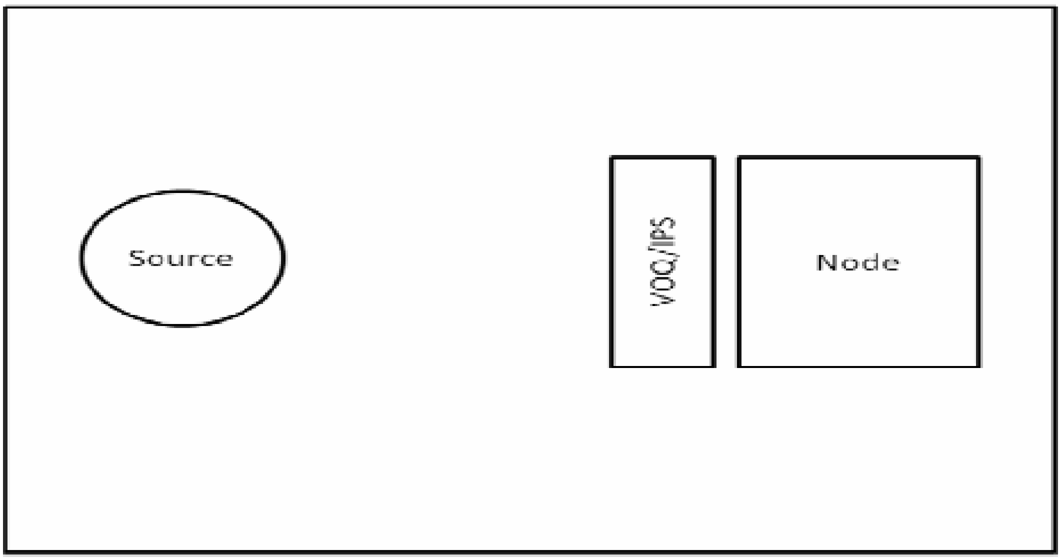

Figure1 Insertion of VOQ and IPS in a network

Thus, it is simpler to insert the VOQ and IPS between the link and the node to model the behavior of IPS and VOQ at the node ingress. A node that had a number of inputs and performed aggregate policing on all of these inputs had to be implemented. The way that the VOQ and IPS work meant that this was not very easy to do. It required a configuration as in Figure 2. In this configuration, all the traffic is multiplexed onto the same link, and then the queuing and scheduling is applied before the packets enter the next node. [1]

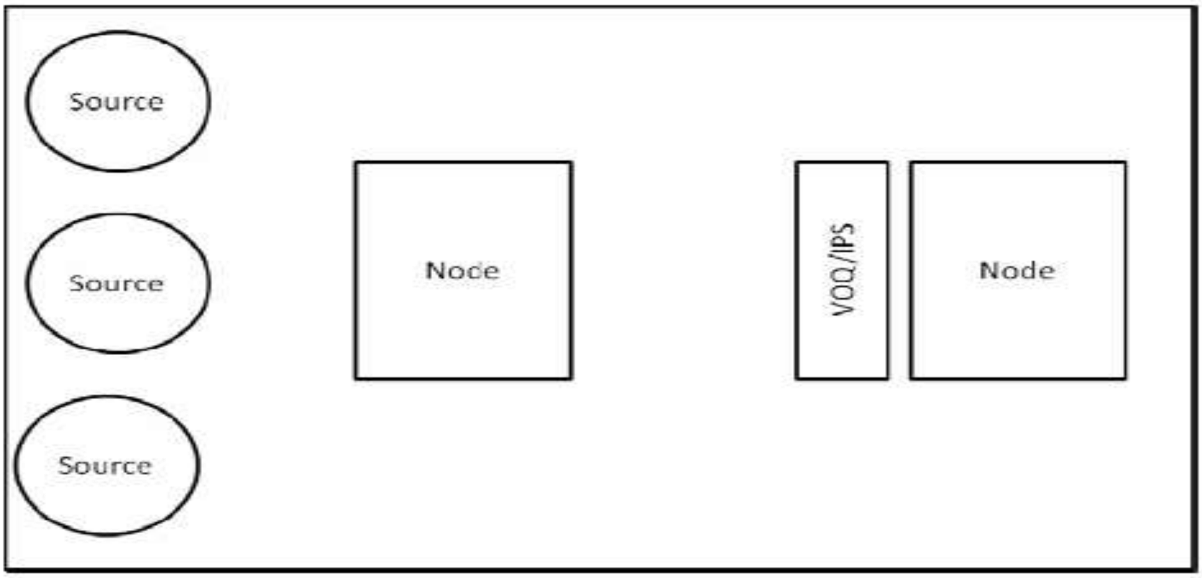

Figure 2 Configuration that enables VOQ and IPS to be used

This configuration is not the most efficient it requires the use of extra nodes and links. An alternative solution is to develop a new type of node that contains VOQ and IPS that can be used to police all the traffic entering the node. This is shown in Figure 3. This requires definition of a new type of node one with a VOQ and IPS installed.

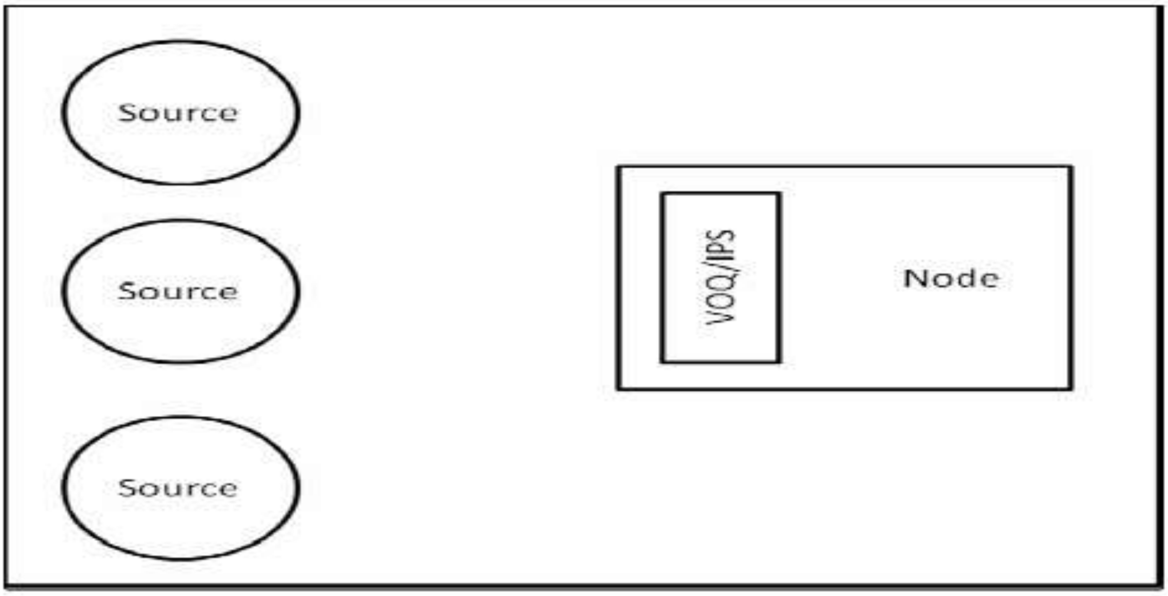

Figure 3 another implementation of VOQ and IPS

To implement this, one should develop a new type of node e.g. called an IPS node. This is a Tcl class that contains an IPS and VOQ as well as the other components and unfamiliarity with ns-2, I was unable implement this type of configuration.

Input queuing has no scaling limitations but it exhibits a performance bottleneck known as Head-of-Line (HoL) blocking. Figure 4 show the avoidance of HoL blocking.

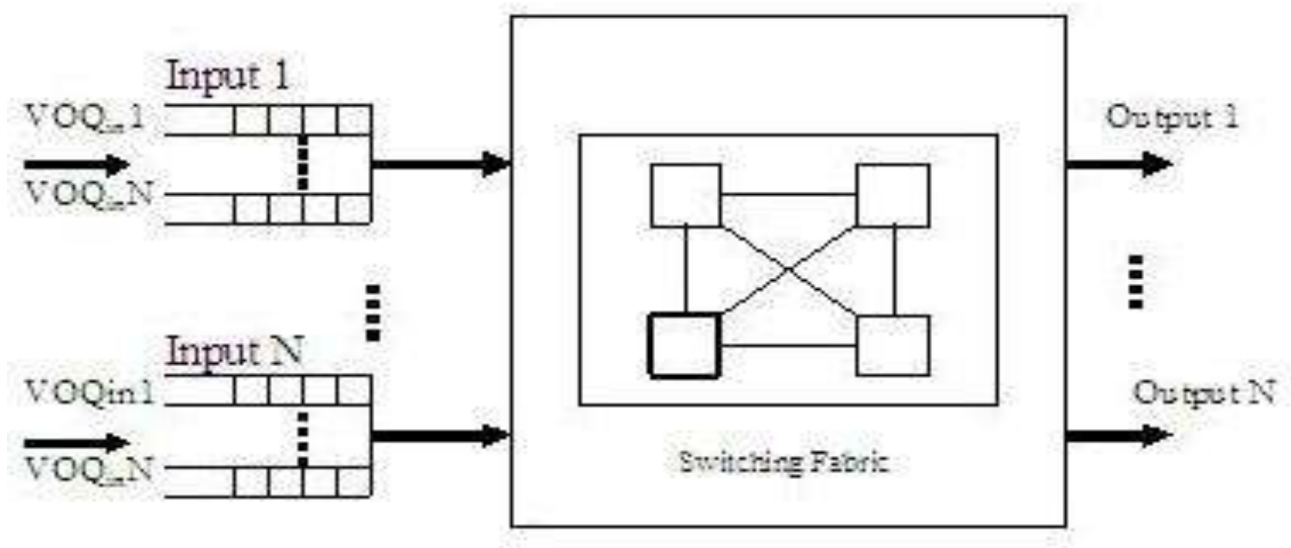

Figure 4 Virtual Output Queuing

VOQ is an input queuing strategy in which each input port maintains a separate queue for each output port. [7][13]

0 ≤ P VOQi,j ≤ 1 for all time slots. The sum ∑ W VOQi,j in (7) above is the total of all HoQ packets retrieved at that time slot, i.e. if k packets have been retrieved and k+1 is being retrieved, the sum will be for those k+1 packets. [1][8]

2. IPSWPP Scheduling Process

IPSWPP has the ability to schedule packets in parallel, which is what makes it more superior than IPS. The whole process of IPS WPP is implemented as shown Program 1.

```
#include<stdio.h>
#include<conio.h>
Void main ()
{
int i, j,n;
long int k, A=2, T=1,sum=0;
int VOQ[10][10],VOQ1[10][10],VOQ2[10][10];
float pmax, P,W[10][10];
clrscr();
printf("Enter Number of Objects in Set Z : \n");
scanf("%d",&n);
k=n;
//pmax=0;
printf("Enter Input Port 1 and Input Port 2:\n" );
for(i=1;i<=k; i++)
{
        for(j=1;j<=n;j++)
        {
                scanf("%d %d",&VOQ1[i][j],&VOQ2[i][j]);
}        }
printf("\t\tWeight of Packet in VOQi,j is Calculate :\n" );
for (i=1;i<=k; i++)
{
            for (j=1;j<=n; j++)
            {
               VOQ[i][j]=VOQ1[i][j]*VOQ2[i][j];
            }
}
for(i=1;i<=k; i++)
{
  for(j=1;j<=n;j++)
  {
       A=A*VOQ[i][j];
       T=T*VOQ[i][j];
      W[i][j]=2*A+T*1;
      sum+=W[i][j];
      P=(W[i][j]/sum); printf("WeightPacketinVOQij[%d][%d]=%.2f\n",i,j,P[i][j]);
    if(P>pmax)
             pmax=P;
 }
}
printf("Probability = %.2f\n",P);
printf("Allocate Packet with Pmax to Worker Processor = %.2f",pmax);
getch();
}
```

IPSWPP adds another loop on of normal IPS. The additional loop is not a problem since there are a few calculations being made per iteration hence the delay caused by calculations is insignificant, what is important is the number of packets that are being scheduled at the same time, which is equal to NP. When the NP = 1, then it is representation of IPS because now it means only one packet will be scheduled per iteration.

3. Simulation Results and Analysis

The performance of IPS WPP with respect to IPS on the two-stage queuing i.e. Multistage Queuing and Scheduling Architecture i.e. Scheduling (MQAS) architecture extensive simulations have been conducted. The queuing occurs at the input and scheduling occurs at the output of the router. The simulation four type statistics:

3.1 Delay of packets for different utilizations:

When the packets are transmitted from their sources, depending on the queue condition/state, the inter arrival rate and the efficiency of scheduling algorithm these packets experience different delays.

3.2 Average delay of packets with multiprocessors:

It depends on the number of processors that how many packets can be processed simultaneously because in routers processors share the processing of packets as they arrive at the input ports.

3.3 Average delay of packets with multiport:

Routers have different number of ports (input and output), as the number of input ports increase the arrival of packets to the router will also increase, this leads to the growth of queue length.

3.4 Throughput:

This represents the total number of packets delivered as a fraction of total packets sent. As packets arrive at the input ports, the queues at the inputs will grow larger.

4. Delay of Packets with Different Utilizations

Figures 4 to 8 shows a plot of latency against packet number and this shows the delay experienced by each packet for different utilizations, 10%, 30%, 60%, 80%, 100% when the number of ports, both input and output, N = 5.

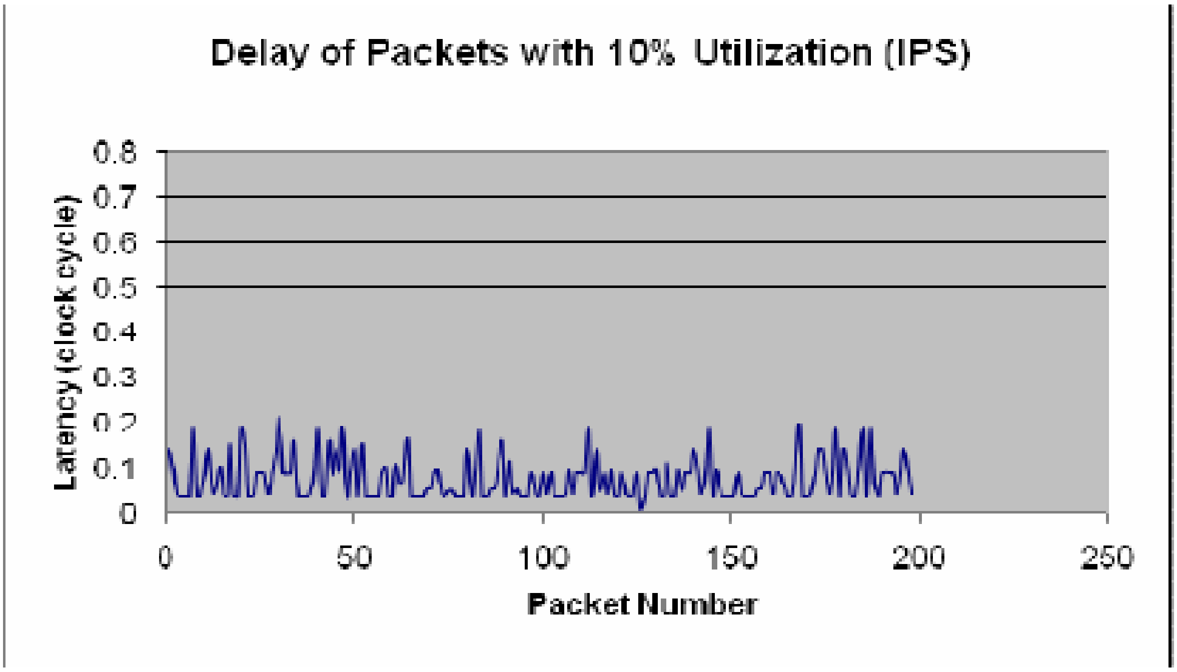

Figure 4(a) IPS Scheduling Algorithm

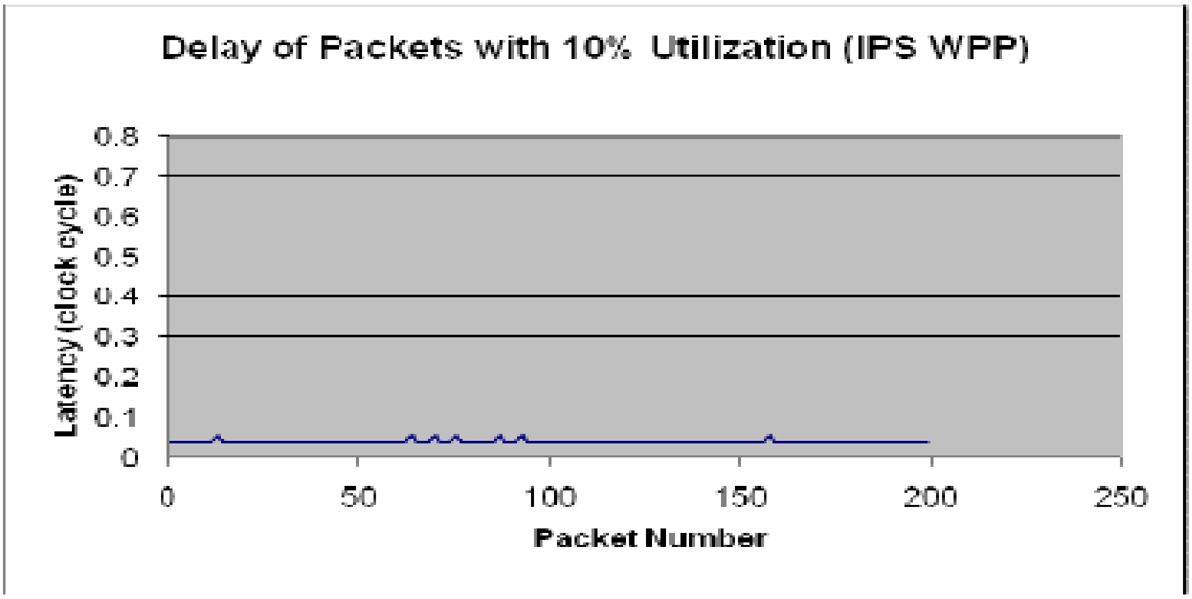

Figure 4(b) IPS WPP Scheduling Algorithm

Figure 4 Delays of Packets at 10% Utilization

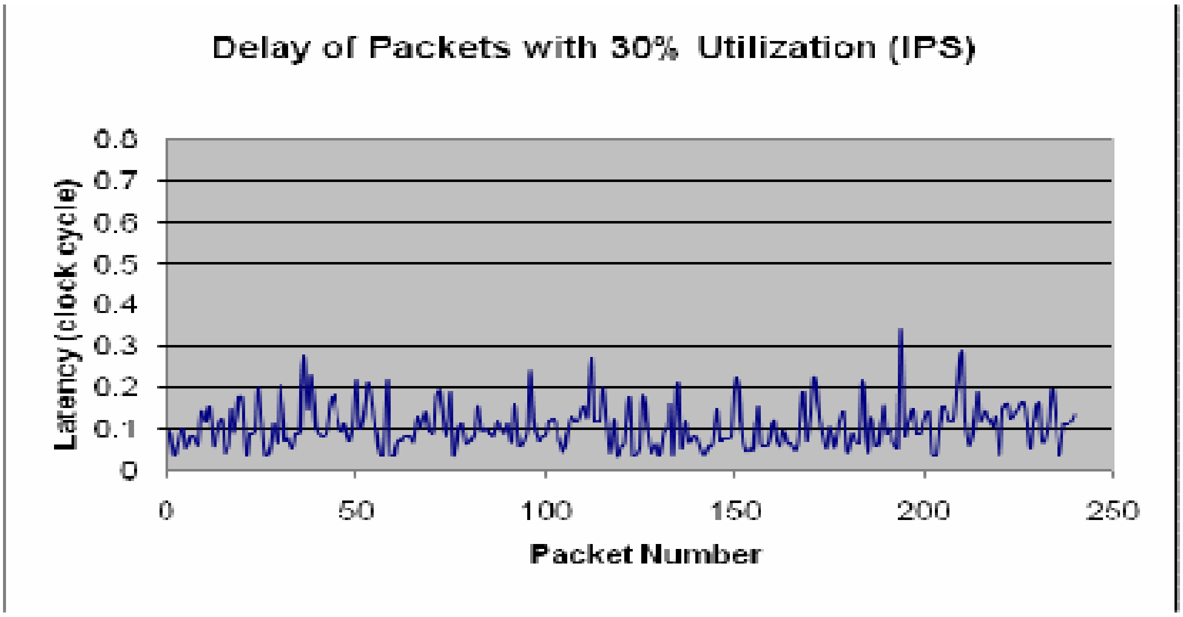

Figure 5(a) IPS Scheduling Algorithm

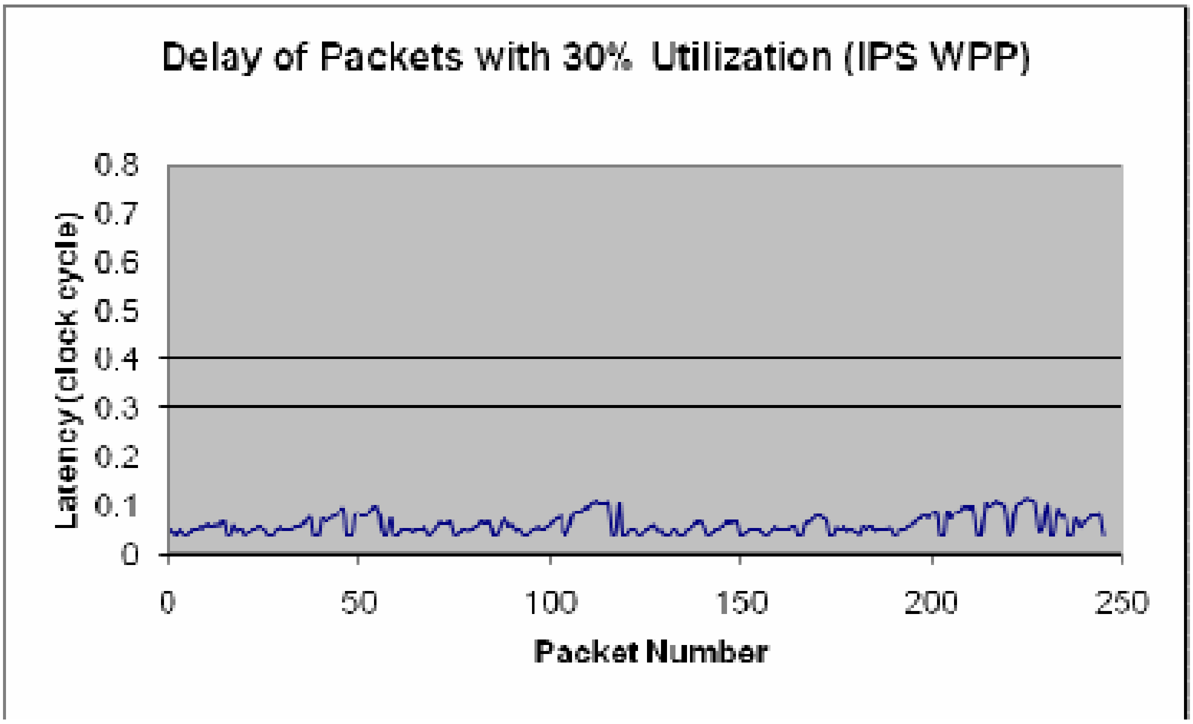

Figure 5(b) IPS WPP Scheduling Algorithm

Figure 5 Delays of Packets at 30% Utilization

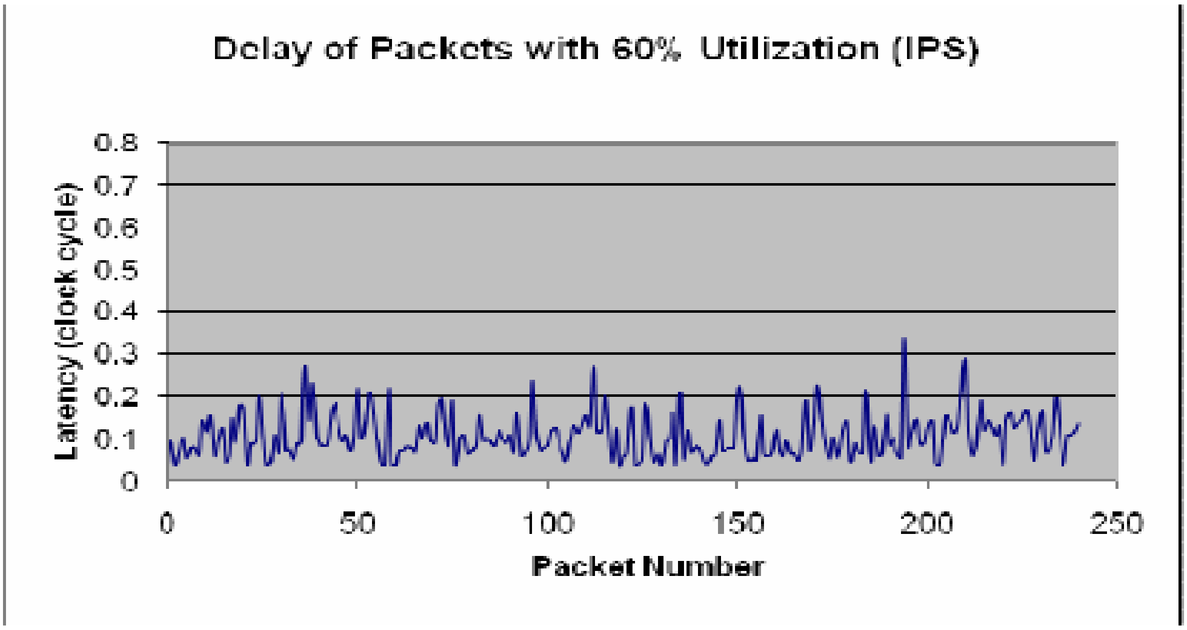

Figure 6(a) IPS Scheduling Algorithm

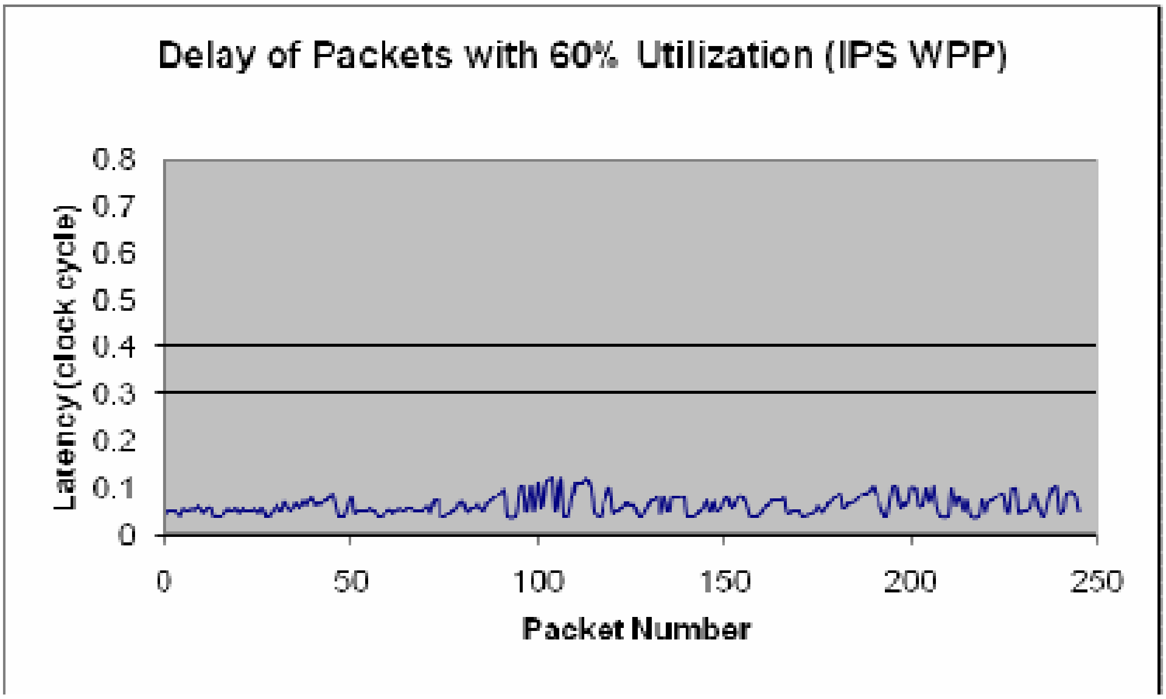

Figure 6(b) IPS WPP Scheduling Algorithm

Figure 6 Delays of Packets at 60% Utilization

`

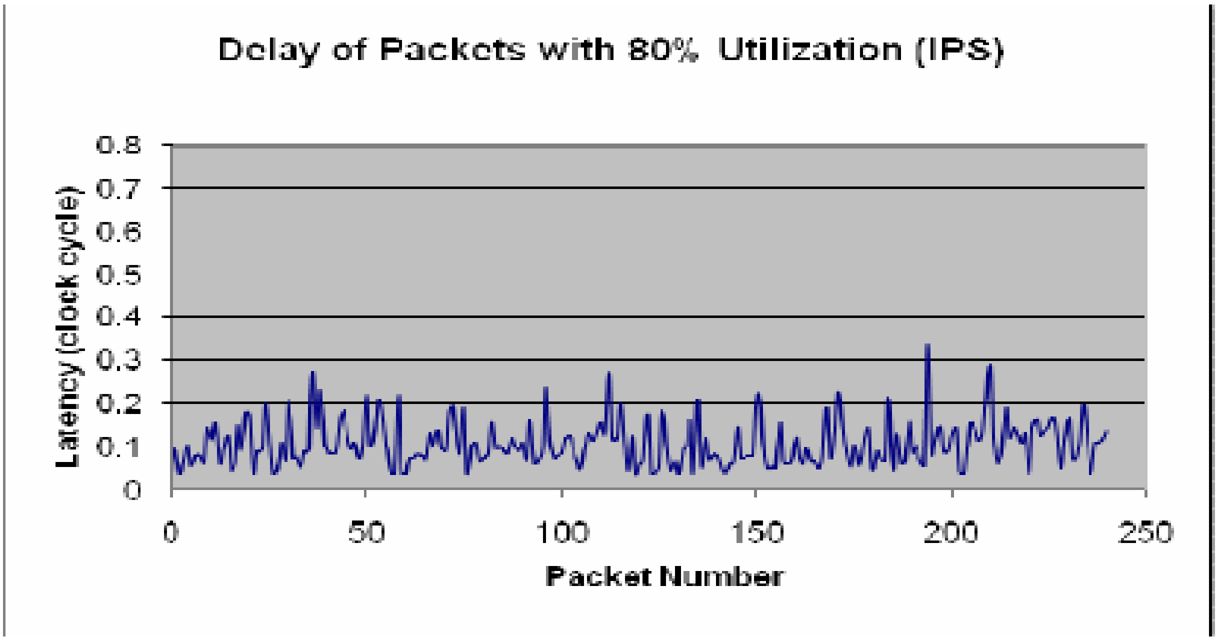

Figure 8(a) IPS Scheduling Algorithm

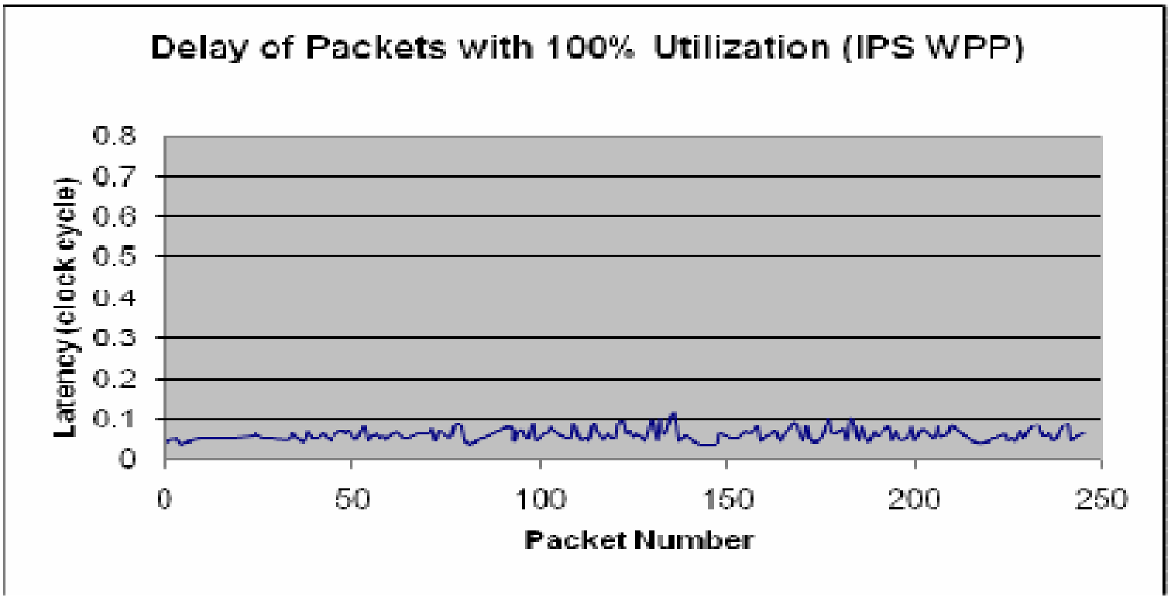

Figure 8(b) IPS WPP Scheduling Algorithm

Figure 8 Delays of Packets at 80% Utilization

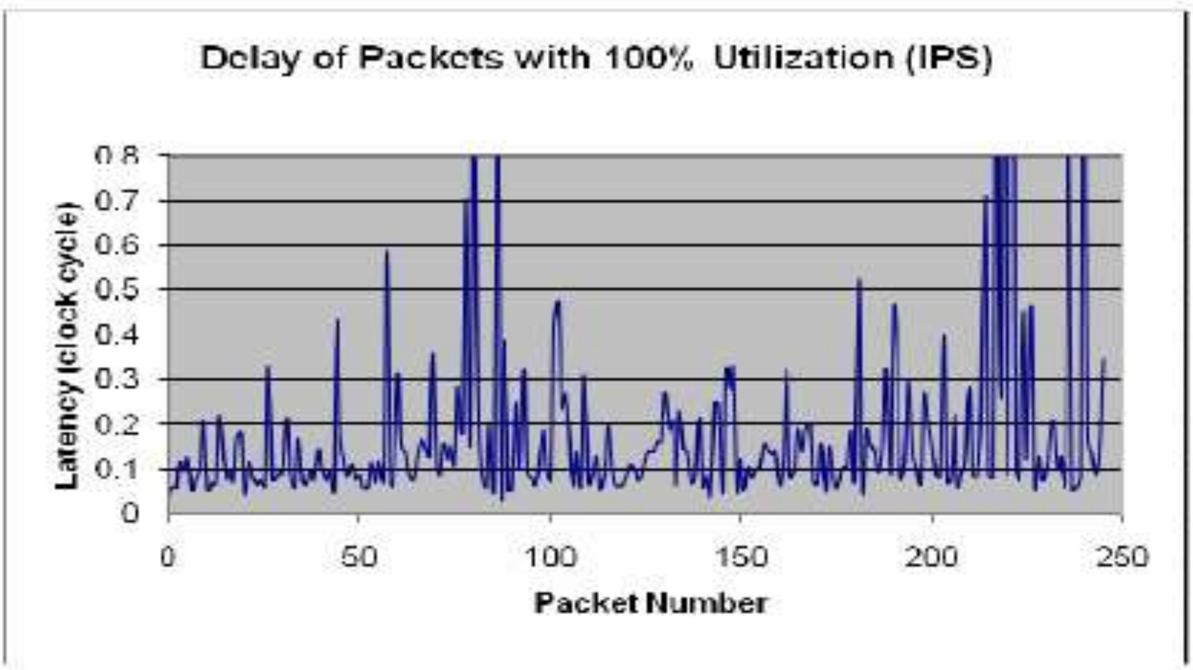

Figure 8(a) IPS Scheduling Algorithm

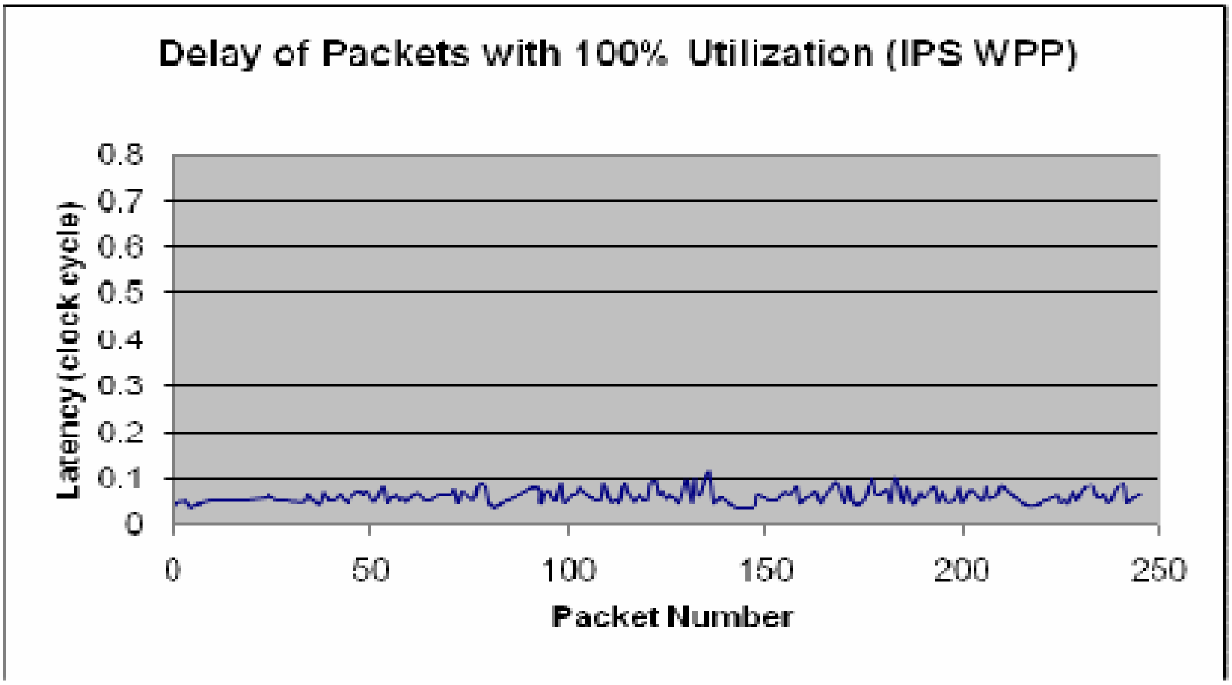

Figure 8(b) IPS WPP Scheduling Algorithm

Figure 8 Delays of Packets at 100% Utilization

One of the aims of the IPS is to minimize packet delay as much as possible. Shown in the Figures above are plots of the delay experienced by each packet. The graphs do not show a constant increase or decrease in delay because some packets arrive at the queues which are empty and hence do not experience any long delays. It also shows that the busy queue with large number input ports cause longer delays.

Looking closely, at low utilizations (10%), delay of packets is almost constant in IPS WPP and as utilization increases delay starts of as almost being constant and increases slightly when it gets to the packets which have a larger sequence number (arrive late).

5. References

[1] Yu-Kwong and Ishfag Ahmad, ”Parallel Algorithm for compile- Time Scheduling of Parallel Programs on Multiprocessors”, University of Hong-Kong.

[2] A. Demers, S. Keshav, and S. Shenker, “Analysis and simulation of a fair queuing algorithm”, Journal of Internetworking Research and Experience, 1(1):3–26, September 1990.

[3] A. Demers, S. Keshav and S. Shenker, “Analysis and simulation of a fair queueing algorithm”, ACM SIGCOMM ’89, vol. 19, no. 4, pp. 3-12, Austin, TX, Sept. 1989.

[4] H. Adiseshu, G. Parulkar, and G. Varghese, “A reliable and scalable stripping protocol,” SIGCOMM, pp. 131–141, 1996.

[5] Darren Spohn, Tina Brown and Scott Grau, ” Data network Design”, third edition published by Tata McGraw-Hill Companies, 2002.

[6] C.H. Papadimitriou and K.Steightz,Combinational Optimization, “Algorithm and Complexity”, Prentice Hall, Englewood Cliffs,1982.

[7] G. Nong and, 62–69 M. Shreedhar and G. Varghese, “Efficient fair queuing using deficit round robin”, IEEE/ACM Trans. Networking, vol. 4, no. 3, pp. 375-385, Jun. 1996.

[8] O. Salami and H Anthony Chan, “A Scheduling Scheme to Improve QoS Provisioning for IP traffic”, University of Cape Town.

[9] Mohsen Bayati ,Balaji Prabhakar, Devavrat shah and Mayank Sharma,” Iterative Scheduling Algorithms”, IEEE INFOCOM, 2007.

[10] N.McKweon. Scheduling Algorithms for Input-Queued Cell switches. Ph.D, dissertation, Dept. Elect. And Comp. Eng. , University of California at Berkeley, 1995.

[11] N. Shah, “Understanding network processors”, Master’s Thesis, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, 2001

[12] Iliadis, W.E. Denzel, “Performance of packet switches with input and output queuing”, IEEE ICC’90, Atlanta, GA, pp. 747-53, Apr. 1990.

[13] D. Pan and Y. Yang, “Pipelined two step iterative matching algorithms for CIOQ crossbar switches,” Proc. of ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS), Oct. 2005, Princeton, NJ.

[14] M. Karol, K. Eng, and H. Obara, “Improving the performance of input-queued ATM packet-switching”, in Proceedings IEEE INFOCOM'92, pp. 110.115, 1992

[15] The Network Simulator ns-2, Available HTTP: http://www.isi.edu/nsnam/ns/.