Probabilistic Scheduling of IP traffic.

Oladayo Salami, *Member,IEEE,* H Anthony Chan, *SrMember, IEEE* and Mqhele E. Dlodlo,
SrMember, IEEE.

Abstract—The co-existence of different traffic with quality of service (QoS) requirements limits the scalability of queuing and scheduling strategies. Two basic single stage queuing strategies implemented in routers are Output Queuing (OQ) and Input Queuing (IQ). OQ offers optimal throughput and guarantees QoS but it is not scalable. Even though IQ is simple and has been identified as the most scalable, the optimum throughput achievable is 58.6% due to Head of Line (HoL) blocking. Virtual Output Queuing (VOQ), a proffered solution to HoL blocking can achieve 100% throughput with an effective scheduling algorithm. Most of the existing schedulers for VOQ are deterministic algorithms which either guarantee QoS or do not. However, the deterministic algorithms that guarantee QoS have high time complexity. This paper proposes a probabilistic scheduling algorithm known as Iterative Probabilistic Scheduling (IPS) for the VOQ. The queuing architecture presented is a multi-stage queuing and scheduling (MQAS) in which VOQ is implemented at the input port and OQ at the output port of the router.

Index Terms—Iterative Probabilistic Scheduling (IPS), Quality of Service (QoS), Queuing, Scheduling, Virtual Output Queuing, (VOQ).

I. INTRODUCTION

THE surge in end user's traffic within today's networks demand an increase in the bandwidth provided by these networks. These demands in network capacity are met by the technological advancement that has evolved since the 1990's. For example new generations of Ethernet are operating at 1Gbps, emerging from 10Mbps. During the early 2000, evolving next generation networks and heterogeneous networks like 4G targeted speeds of 70-200Mpbs. In addition, large ISPs began to use OC-192 circuits that transmit at approximately 10Gbps. Also, optical switching technologies with line rate of 10-40Gbps are being developed to meet the increasing demand in bandwidth [1].

Different applications from end user's of these networks constitute packets which generate varying traffic patterns. Consequentially, mission critical traffic can coexist within a network thereby posing a challenge to the design of network elements. The challenge is that packet processing functions of the network elements need to provide QoS for different traffic types at line speed.

Packet processing functions, the fundamental processing operations of the router on a packet include policing, classification, queuing, shaping, and scheduling [2].

These challenges have raised two questions for the router designers. The first question deals with whether these packet processing functions should be performed at the input or at the output of the router for optimal throughput

The second is how to implement the above functions to scale with increasing network speed and while provisioning QoS for end user's traffic.

This paper presents a multistage queuing and scheduling (MQAS) architecture as a solution to the queuing and scheduling challenges. MQAS is a two-stage architecture which implements Virtual Output Queuing (VOQ) at the input ports and First-In First-Out (FIFO) queuing at the output ports of the router. The scheduling algorithm for the VOQ proposed in this paper is the Iterative Probabilistic Scheduling (IPS). The paper also provides an analysis of some deterministic QoS guaranteeing scheduling schemes.

The organization of this paper is as follows; section 2 gives an overview of the QoS architecture. Queuing and Scheduling issues are discussed in section 3. Analytical and simulation results are presented in section 4. Section 5 presents the analytical model of our proposed Queuing Architecture and Scheduling Algorithm while section 6 gives a conclusion.

II. QOS ARCHITECTURE

Quality of service provisioning is implemented on a hierarchical architecture with three levels [3]. The hierarchy of the architecture is outlined below:

1) The first level provides QoS within a single network element through classification, queuing, scheduling, and traffic-shaping techniques.
2) At the second level, QoS signaling i.e. admission control and resource reservation are performed from end to end and between network elements.
3) The third level provides QoS policy, management, and accounting, to control and administer end to end traffic across a network.

The focus of this paper is on the first level of the above mentioned architecture.

The functions performed on the first level of the QoS architecture are known as packet processing functions. Without effective packet processing functions, it is almost impossible to deliver meaningful service guarantees during network congestion [2].

QoS guarantees are ensured through the order in which

Manuscript received August 8, 2006. This work was supported in part by the Broadband Center of Excellence with funding from Telkom, Siemens, and National Research Foundation, South Africa.

Miss Oladayo Salami is with the University of Cape Town, Rondebosch, Cape Town, 7701,South Africa (e-mail: oladayo@crg.ee.uct.ac.za).

H. Anthony Chan is a professor in the Department of Electrical Engineering, University of Cape Town, Private Bag, Rondebosh 7701, South Africa (e-mail: h.a.chan@ieee.org).

Mqhele E. Dlodlo is a professor in the Department of Electrical Engineering, University of Cape Town, Private Bag, Rondebosh 7701, South Africa (e-mail: mdlodlo@ebe.uct.ac.za).

the packet processing functions are carried out. As packets arrive at the ingress policing ensures that they maintain the agreed service level guarantee. Classification sorts the packet according to their QoS requirement while queuing buffers these packets in different queues within the router. Shaping ensures that the packet meets the requirement of the downstream network and do not cause congestion. Scheduling determines the optimum order to forward these packets. These functions can be implemented either in software or hardware [1].

III. QUEUING AND SCHEDULING

A queue is where the packet waits from its arrival time to its service time. Queuing algorithms provide strategies for buffering packets according to classification. Some examples of queuing algorithms are Priority Queuing (PQ), First In First Out (FIFO), and Custom Queuing (CQ) [4][5][6].

One important issue affecting the routers' scalability is where to implement a queue strategically for optimal performance in the router [7]. Two major queuing strategies are 1) output queuing 2) input queuing. These queuing strategies are discussed below.

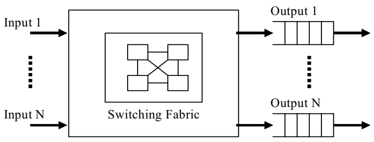

Fig. 1. Output Queuing

The Output queuing (OQ) strategy is shown in Fig. 1. Here packets are buffered at the output ports to maximize router throughput. However, if packets destined for the same output port arrive simultaneously, then the buffer will have to queue these packets at a higher speed than the line speed. The required high speed places a scaling limitation on the router.

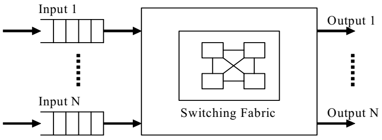

Fig. 2. Input Queuing.

Input queuing (IQ) is as illustrated in Fig. 2. Packets are buffered at the input ports in this strategy and only the first packet in any queue is eligible for transmission at a time. IQ has no scaling limitations but it exhibits a performance bottleneck known as Head-Of-Line (HOL) blocking. HOL blocking happens when a packet at the head of a queue is blocked thereby preventing other packets behind it from being transmitted. It has been shown that HOL blocking reduces throughput to as low as 58.6% [4][7][8].

With the advantages and drawbacks of the queuing strategies mentioned, the simplest and most scalable approach is the IQ. However, IQ has a draw back of the HOL blocking effect. To avoid HOL blocking, Virtual Output Queuing (VOQ) in Fig. 3 was proposed [4][8]. VOQ is an input queuing strategy in which each input port maintains a separate queue for each output port. It has been shown that VOQ can achieve a 100% throughput performance with an effective scheduling algorithm. This algorithm should be able to provide a high speed mapping of packets from inputs to outputs on a cycle-to-cycle basis [4].

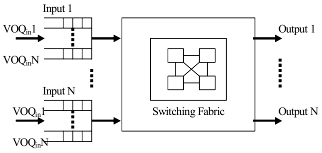

Fig. 3. Virtual Output Queuing.

Scheduling is a major component in QoS provisioning within the router [9]. It is an algorithm which determines the order in which packets are served. The algorithm must be simple, fair and prevent the starvation of any packet in a queue [10]. However the VOQ architecture has made these requirements more complex to achieve.

Packet scheduling in VOQ is similar to the bipartite graph-matching problem (BMP) which tries to find a conflict-free pairing of inputs to outputs [4][11]. The scheduler must retrieve the state information of all contending packets and perform a maximal matching of these packets to an output port. In addition, it must be able to arbitrate fairly and guarantee quality of service among these packets under uniform and non uniform traffic.

QoS guarantee is usually characterized in terms of bounds on bandwidth, delay, jitter, packet loss rate, or a combination of these parameters. These guarantees can be deterministic or probabilistic. Scheduling algorithms which can also be deterministic or probabilistic guarantees QoS accordingly [12].

Deterministic VOQ scheduling algorithms allows the prediction of future packet selection, but have been shown to be slower, difficult to provide and underutilizes network resources [13]. Some of these algorithms are able to guarantee QoS while others do not. The major scheduling algorithms for the VOQ are deterministic algorithms and are outlined in table I.

TABLE 1
VOQ DETERMINISTIC SCHEDULING ALGORITHMS

ALGORITHM	TIME SLOT ASSIGNMENT	MAXIMAL MATCHING	STABLE MATCHING
Time complexity	$0(N^{2.5})$	$0(N^2)$	$0(N^2)$
Maximum throughput	100%	50%	50%
Differentiated service	Not supported	Not supported	supported
Best supported traffic	CBR	CBR	CBR and VBR

QoS guaranteeing deterministic algorithms are limited in performance because they are traffic pattern dependent, do not provide fairness and have high time complexity. They are not scalable for use in networks carrying non-uniform traffic. Other deterministic algorithms are the Parallel Iterative Model (PIM) with O(logN) complexity and SLIP which is the simplest and fastest of all these deterministic algorithms mentioned [11]. iSLIP is a variation of the PIM algorithm [14]. Although PIM and iSLIP have lower time complexity, they do not guarantee QoS. The major reason for their inability to guarantee QoS can be attributed to their random and round robin scheduling pattern. In addition, these algorithms are too complex for hardware implementation.

Conversely, probabilistic algorithms are able to capture the needs of network applications, all of which can tolerate some delay. In view of these, for the VOQ section of the MQAS architecture we propose an Iterative Probabilistic scheduling (IPS). In the next section, we have analyzed and compared the performance of the FIFO, PIM, and iSLIP.

IV. ANALYTICAL AND SIMULATION RESULTS.

FIFO, PIM, and i-SLIP have been chosen for analysis in order to demonstrate the limitations of FIFO and improvement provided by PIM and i-SLIP using the VOQ strategy. Simulations of these algorithms were carried out under a variety of input load or utilization (U) to get the average queuing latency (L) results for different router sizes. Incoming packets were assumed to be an independent, identically distributed (i.i.d.) Bernoulli process with destinations uniformly distributed over all output ports.

Fig. 4 shows the FIFO Latency-Utilization (L-U) curve for routers with different number of ports (N).

The legend in fig.4 applies to fig.5 to fig.7. Table II summarizes the maximum input load (U) that can be handled by FIFO-enabled router before saturation. Fig.4 shows that for very large routers with N>8 the maximum throughput (U) that can be offered is approximately 58.58%. This limitation is as a result of the Head of Line (HOL) blocking effect. Therefore, for routers with large N, the performance of FIFO is limited to 58.58%. The simulation also shows that the 58.58% utilization is asymptotic with N, where small N gives asymptotes above 60%.

The L-U curve for the PIM algorithm with a single iteration (PIM-1) in fig.5 shows a poor utilization of the router as N increases. PIM-1 gives only a maximum utilization of 63% for routers with N>8. This is a slight increase to the 58.58% offered by FIFO. However, PIM with four iterations (PIM-4) gives a significant improvement by reducing latency and thereby increasing the utilization of the router. PIM-4 remains stable with an offered input load in excess of 95% as shown in fig. 6.

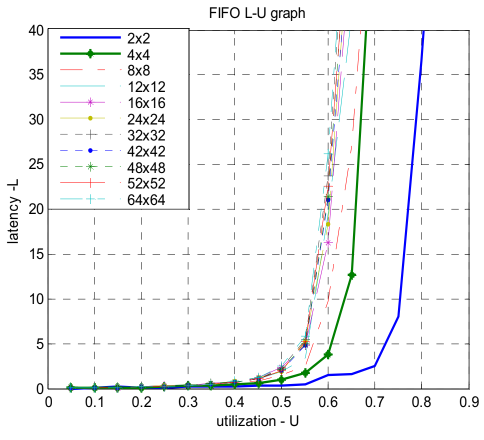

Fig. 4. FIFO Latency-Utilization Curve

TABLE 2
FIFO PERFORMANCE FOR DIFFERENT ROUTER SIZES

N	U
2	0.75
4	0.6533
8	0.5990
12	0.5858
16	0.5858
∞	0.5858

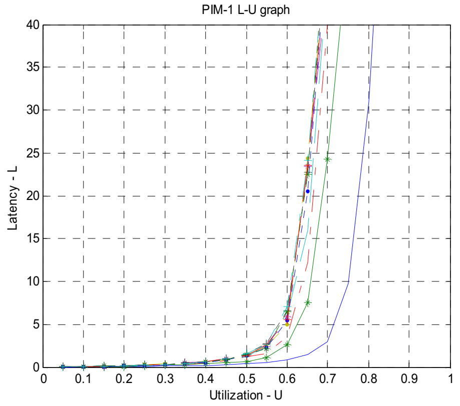

Fig. 5. PIM-1 Latency-Utilization Curve

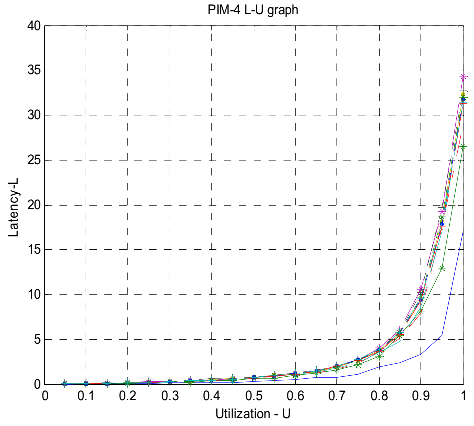

Fig. 6. PIM-4 Latency-Utilization Curve

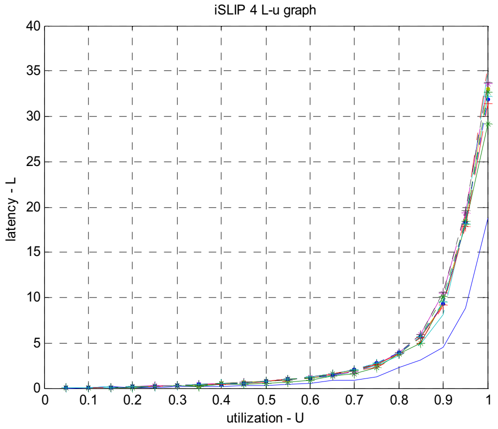

Fig. 7. iSLIP Latency-Utilization Curve

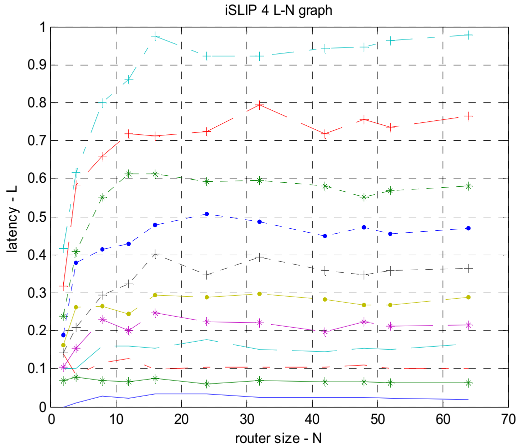

Fig. 8. iSLIP Latency – Router size Curve

Fig. 7 gives the L-U curve for i-SLIP, simulated with four iterations because the latency for each router size reduces with the number of iterations [11][14]. Also, the algorithm gave the same result for iterations ≥4 for all router sizes meaning that the algorithm becomes stable with four iterations.

Unlike FIFO and PIM, the performance of i-SLIP fluctuates as the router size increases. It can be shown in fig. 7 that for all router sizes, the average latency approaches a constant for low utilization (input load). This fluctuation is detailed in fig.8, as N increases latency decreases with utilization increases. The different colours in fig 8 represent utilization from 5% to 55%. For small router sizes increasing N increases latency and decreases utilization while for large router sizes increasing N decreases latency and increases utilization as shown in table III.

From the simulations, although PIM and iSLIP which are deterministic algorithms have high performance, they do not guarantee QoS. These algorithms only perform well under uniform traffic and are therefore not scalable for use in networks with non-uniform traffic.

TABLE 3
I-SLIP PERFORMANCE FOR DIFFERENT ROUTER SIZES

N	U
2	0.952
4	0.915
8	0.900
12	0.900
16	0.885
24	0.890
32	0.875
42	0.890
48	0.887
52	0.920
64	0.923

V. PROPOSED QUEUING AND SCHEDULING ARCHITECTURE.

To alleviate the queuing and scheduling challenges, this work is implementing a multistage queuing and scheduling (MQAS) strategy in Fig. 9. MQAS is a two-stage queuing and scheduling architecture. It combines the performance of output queuing (OQ) with the scalability of the VOQ [15]. The VOQs in the first stage of this architecture maintains a separate queue at the input ports for each output port [11][14].

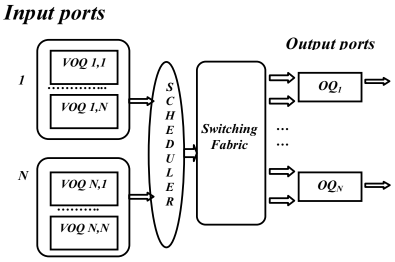

Fig. 9. Multistage Queuing and Scheduling Architecture (MQAS).

These virtual output queues are implemented in the memory buffers at each input port. That means for an N-output port router, each input port maintains N VOQs in its memory buffer, as indicated in Fig. 9. Therefore, for a router with N-input ports and N-output ports, the total number of virtual output queues to be maintained will be N^2. The total number of memory buffers implemented at each input is N.

The second stage of MQAS uses an OQ strategy in which packets are queued on a FIFO basis. One queue is maintained per output port, and the scheduler also uses FIFO to match packets to the output link. At the output, packets experience minimum delay because there is no HOL blocking so that the throughput is maximized [15]. QoS is also guaranteed at these ports because contention is already resolved at the input ports.

A. *The Iterative Probabilistic Scheduling (IPS).*

The Iterative Probabilistic Scheduling (IPS) implemented at the VOQ stage of the MQAS is a probabilistic algorithm. IPS is proposed for the VOQ stage to improve on the implementation complexity of i-SLIP.

IPS focuses on two quality of service parameters, which are delay and bandwidth. The scheduling pattern and the QoS guarantee are probabilistic. The aim of IPS is to provide fairness to all traffic types while guaranteeing QoS. This makes IPS scalable for implementation in all networks. Contention is made stringent and fair with the weights assigned to the QoS parameters.

IPS orders packets based on an estimation of the bandwidth required for transmission and of the waiting time. The weight of each packet is calculated using these two parameters which are retrieved during the scheduling process. IPS then determines the probability of transmitting each packet retrieved during a time slot. The packet with the highest probability is assigned the Highest Bandwidth Packet (HBWP) tag and transmitted to its corresponding output port. Only packets with the HBWP tag are served during a scheduling process [16].

1) *The Model.*

Consider a router with N input ports and N output ports implementing MQAS architecture. Assume an M/M/1 queue model: The first 'M' indicates that the arrival rate (λ) of packets is Poisson distributed, the second 'M' indicates an exponentially distributed service rate (μ), and '1' implies that the router is a single server system.

With MQAS architecture, it can be deduced that each input port has N virtual output queues (VOQ). Let ‘i’ represent an input port and ‘j’ an output port within the router. Therefore, $VOQ_{i,j}$ denotes a VOQ in input port ‘i’ queuing packets for output port ‘j’, and OQ_j represents an output queue in output port ‘j’.

For all non-empty queue at the input the weight of a packet in $VOQ_{i,j}$ is given by:

$$WP_{VOQi,j} = eBW_{VOQi,j}*2 + eQ_{VOQi,j}*1, \quad (1)$$

$eBW_{VOQi,j}$ is the estimated transmission bandwidth given by the size of the packet. $eQ_{VOQi,j}$ is the estimated waiting time given by subtracting the current time from the last service time of the queue buffering the packet [16]. The probability of transmission of this packet is given by:

$$P_{voqi,j} = \frac{WP_{voqi,j}}{\sum_i WP_{voqi,j}} \quad (2)$$

$0 \le P_{VOQi,j} \le 1$ for all time slots. $\sum WP_{VOQi,j}$ in equation 2 is the total $WP_{VOQi,j}$ of all packets already retrieved at that instant. For example, if two packets have been retrieved and a third is being retrieved, the $\sum WP_{VOQi,j}$ is for these three packets. The $WP_{VOQi,j}$ of other packets are not considered until their parameters are retrieved.

The Iterative Probabilistic Scheduling algorithm operates in three stages which are explained in the next subsections.

2) The Initialization Process of the IPS Algorithm.

At a time slot, input ports with packets in their $VOQ_{i,j}$ send requests for transmission ($REQ_{i,j}$) to corresponding OQ_j. If an input port has packets in all its VOQs it sends a request to each output port as shown in Fig. 10.

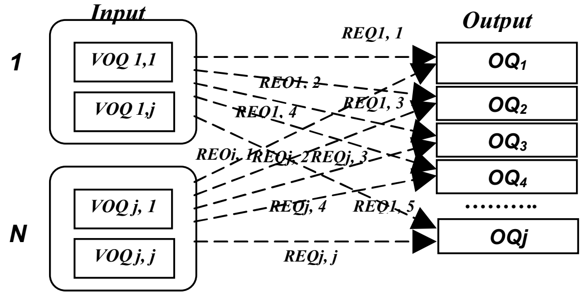

Fig. 10. Input Requests to Output.

1) IPS iterates only through the output queues with requests made to them and chooses an output queue (OQ_j) to serve a packet.
2) It forms a set {Z} i with $REQ_{i,j}$ to OQ_j such that for any j chosen at a time slot all elements in {Z} must be ≤N
3) IPS grants only the request of one input per time using the algorithm described in the next subsection.

3) Operation of the IPS Algorithm.

The flow chart in Fig.11 outlines the operation of the IPS algorithm. The calculation of $P_{VOQi,j}$ in the flow chart is subject to the normalizing condition: $\sum P_{VOQi,j} = 1$

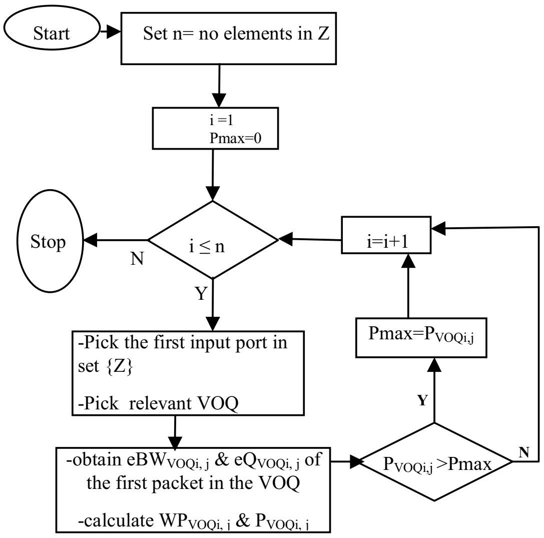

Fig. 11. Flow Chart of IPS algorithm.

4) Scheduling Process of the IPS Algorithm.

After retrieving the state of all packets contending for transmission, IPS performs the following process to schedule a packet to send to OQ_j

STEP 1: Pick the packet with the maximum $P_{VOQi,j}$.

STEP 2: Append the (Highest Probability Packet) HPP tag to the packet.

STEP 3: Transmit the packet to chosen OQ_j.

At any scheduling time slot, only the packet appended with the HPP tag is scheduled.

At the output port of the MQAS architecture, packets are mapped to the output link on the basis of FIFO. There is an increased throughput at the output ports because no packet major packet processing is carried out.

A summarized pseudo code of the operation of the Iterative Probabilistic Scheduling algorithm is as below:

```
1. Insert all inputs with REQ into
   an Array[Z]
2. iterate to choose OQ
3. Set n=number of objects in
   Array[Z]
4. Set Pmax=0
5. Set counter: for( i=1, i<n, i++)
6. Picks the first object in Array
   [Z].
7. get the Head of Queue packet of
   VOQ in this input port
9. calculate WP and P
10.If P > Pmax
11. Pmax=P }
12.Append HPP to packet with Pmax
13.Transmit packet.
```

A worst-case scenario occurs if all inputs sends a request. In this situation, steps 5 to 11 are executed a maximum of N times, where N is the number of input ports. The number of execution times of each step is analyzed in table 4. Statements 2, 3, 4, 11, 12 and 13 are executed once; therefore the total computing time is 7n+5. From the “Big-Oh” theorem: Let n be a non-negative integer parameter describing t he number of request made by the input ports.

TABLE 4
IPS TIME COMPLEXITY ANALYSIS

Statement	No of times executed
1	n
2	1
3	1
4	1
5	n
6	n
7	n
8	n
9	n
10	n
11	n
12	1
13	1
T(n)	7n+5

Let $f(n)$ be a function describing the performance of the algorithm. The performance is in terms of the number of steps that need to be executed multiplied by the number of times they must be re-executed. Let $g(n)$ be another function that represents an upper bound on $f(n)$. If there is a real constant $c > 0$ and an integer constant $n0 \geq 1$, such that $f(n) \leq c.g(n)$ for every integer $n \geq n0$, then $g(n)$ is the "order" of the algorithm. The order of the algorithm is written in "Big-Oh" notation as $O(g(n))$.

Definition: Let $f(n)$ and $g(n)$ be two functions $f(n) = O(g(n))$ or $f = O(g)$ (Read as "f of n is big oh of g of n" or "f is big oh of g") if there is a positive integer C such that $f(n) <= C * g(n)$ for all positive integers n. In the case of IPS, the computational timing analysis $T(n) = 7n + 5$. Mathematically, according to the Big-Oh theorem, $f(n) = 7n + 5 = O(n)$. Consider the constant C needed such that $f(n) <= C * n$ for all n. If $C = 7$, this will not work because $7n + 5$ is not less than $7n$, therefore C has to be at least 12 to cover all n. If $n = 1$, C has to be 12, but C can be smaller for greater values of n (i.e., if $n = 100$, C can be 5). Since the chosen C must work for all n, then C must be 12 so that $7n + 5 <= 7n + 5n = 12n$.

Therefore $T(7n + 5) = O(n)$. From the final equation, intuitively, the running time grows linearly as n grows.

VI. CONCLUSIONS.

The IPS algorithm executes all operations with $O(n)$ time complexity. Therefore it is fast to implement in a high-speed network. In addition, it is a simple algorithm which provides guaranteed service. IPS also ensures fairness and prevents starvation of packets through the weight assigned to the parameters eBW and eQ. As a result, no packet is left indefinitely without being served. Also, IPS is a work conserving scheduler because it continues to serve packets as long as input ports keep sending requests to the output ports. The analytical model of the IPS presented in this paper shows that it can improve on the limitations of the FIFO and the implementation of deterministic schemes such as the iSLIP and the PIM which do not guarantee QoS.

From the analysis, IPS is able to improve on the implementation complexity of the iSLIP in routers using the VOQ strategy. The reduction in implementation complexity can be attributed to the probabilistic nature of the IPS.

REFERENCES

[1] D. E. Comer, *Network Systems Design using Network Processors IXP2XXX version*, 1st ed. (New Jersey: Pearson Prentice Hall, 2005).

[2] Dan Decasper, Guru Parulkar, Sumi Choi, John DeHart, Tilman Wolf, and Bernhard Plattner, A scalable, high performance active network node, *IEEE Network*, *31(1)*, January 1999, 8–19.

[3] S. Vegesna, *IP Quality of Service*, ISBN 1578701163, (Cisco Press, 2001).

[4] J. Soldatos, E. K. Vayias, and G. Kormentzas. On the Building Blocks of Quality of Service in Heterogeneous IP Networks, *IEEE Communications Surveys & Tutorials*, 2005.

[5] A. Demers, S. Keshav, and S. Shenker, Analysis and Simulation of a Fair Queuing Algorithm, *Proc. ACM SIGCOMM '89*, 3–12.

[6] I. Stoica, S. Shenker, and H. Zhang. Core-Stateless Fair Queuing. *IEEE/ACM Trans. Net., 11(1)*, Feb. 2003, 33–46

[7] Dan Decasper, Zubin Dittia, Guru Parulkar, and Bernhard Plattner. Router Plugins – a modular and extensible software framework for modern high performance integrated services routers, *Proc. of ACM SIGCOMM 98*, Vancouver, BC, September 1998.

[8] G. Nong and M. Hamdi, "On the Provision of Quality-of-Service Guarantees for Input Queued Switches, *IEEE Commun. Mag.*, *38(12)*, Dec. 2000, 62–69.

[9] I. Faynberg and H.-L. Lu. An Architectural Framework for Support of QoS in Packet Networks. *IEEE Commun. Mag.*, *41(6)*, June 2003, 98–105.

[10] J.Crowcroft, "Scheduling and Queuing Management," www.cl.cam.ac.uk/Teaching/2003/DigiComm2/sched.pdf.

[11] N. McKeown and T. E. Anderson. A Quantitative Comparison of Iterative Scheduling Algorithms for Input-Queued Switches. *Comp. Networks and ISDN* Sys*.*, *30* 1998, 2309–26.

[12] D.Stiliadis, A.Varma, "Providing Bandwidth Guarantees in an Input-Buffered Crossbar Switch". http://www.bell-labs.com/user/stiliadi/matching.ps.Z.

[13] Bandwidth and QoS Principles available on: http://www.cisco.com/en/us/products/ps6385/products_qanda_item0900aecd8040bc19.shtml

[14] N. McKweon. Scheduling Algorithms for Input-Queued Cell Switches. *Ph.D. dissertation, Dept. Elect. and Comp. Eng.*, University of California at Berkeley, 1995.

[15] Oladayo Salami and H. Anthony Chan. Multi stage Queuing and Scheduling of IP traffic built on Network Processors for QoS Provisioning, *Proc.SATNAC2005*, Drakensberg, South Africa, Sept 2005, 64-65.

[16] O. Salami and H. A. Chan. Iterative Probabilistic Scheduling of IP traffic, *Proc. of 3rd IEEE CCNC'06*, Las Vegas, U.S.A., January 2006, 1336-1327

[17] O. Salami and H. A. Chan. Analysis of a Probabilistic QoS Guaranteeing Scheduling Algorithm for Virtual Output Queuing Strategy, *Proc. of 13th International Conference on Telecommunications, ICT 2006*, Madeira, Portugal, May 2006.