PAPER *Special Issue on Outstanding Papers from APCC 2001*

A Buffer Management Mechanism for Achieving Approximately Fair Bandwidth Allocation in High-Speed Networks

Takashi MIYAMURA[†], Takashi KURIMOTO[†], Kenji NAKAGAWA[††], *Regular Members*,
Prasad DHANANJAYA[††], *Nonmember*, Michihiro AOKI[†],
and Naoaki YAMANAKA[†††], *Regular Members*

SUMMARY We propose a buffer management mechanism, called V-WFQ (Virtual Weighted Fair Queueing), for achieving an approximately fair allocation of bandwidth with a small amount of hardware in a high-speed network. The basic process for the allocation of bandwidth uses selective packet dropping that compares the measured input rate of the flow with an estimated fair share of bandwidth. Although V-WFQ is a hardware-efficient FIFO-based algorithm, it achieves almost ideal fairness in bandwidth allocation. V-WFQ can be implemented in the high-speed core routers of today's IP backbone networks to provide various high-quality services. We have investigated V-WFQ's performance in terms of fairness and link utilization through extensive simulation. The results of simulation show that V-WFQ achieves a good balance between fairness and link utilization under various simulation conditions.

key words: *fairness, QoS, Diff-serv, high-speed network, bandwidth management*

1. Introduction

In today's Internet, the sharing of network resources among the many flows is on a best-effort basis. A malicious user of a network may thus be able to get more bandwidth than a well-behaved user. Protecting bandwidth for well-behaved flows from ill-behaved flows is thus important in the Internet environment. One way to stop ill-behaved flows from adversely affecting other flows is to allocate a fair bandwidth to each flow.

Two types of algorithms can be used to achieve fair bandwidth allocation: scheduling-based algorithms and preferential-dropping-based algorithms.

Scheduling-based algorithms (e.g., Weighted Fair Queueing (WFQ) [4] and its variants [5]–[7]) are known to be ideal as mechanisms allocating bandwidth in a fair way and providing guaranteed QoS. In these approaches, however, must be maintained a separate queue for each flow and state is maintained on a per-flow basis, so that hardware efficiency is out of the question. More precisely, WFQ has a computational complexity of $O(\log(n))$, where n is the number of flows currently queued at a router. WFQ is hard to implement in high-speed backbone routers with trunks that carry large numbers of flows.

Core-Stateless Fair Queueing (CSFQ), proposed by Stoica et al. [9], is a well-known preferential-dropping-based algorithm. The CSFQ mechanism achieves almost ideal fairness without using the per-flow states of the core routers. To avoid maintaining the per-flow state for each router, they use a distributed algorithm in which per-flow state is only maintained for edge routers and is not maintained for core (non-edge) routers. An edge router estimates the arrival rate for each flow and this information is carried by a label in each packet's header to the core router, which utilizes this information in deciding whether to discard or queue each arriving packet. Thus, in terms of the core routers, CSFQ is simpler and easier to implement than WFQ. However, the architecture of the edge routers is still complicated. Although it is easy to extend CSFQ to support flows that have different weights, the algorithm is incapable of accommodating situations where the relative weights of flows differ from router to router.

Recently, Cao et al. [10] have proposed Rainbow Fair Queueing (RFQ), which, like CSFQ, may be used to achieve approximately fair sharing. The approach here is to divide each flow into a set of layers, based on rate. At the edge routers, the packets in a flow are marked with layer labels rather than with explicit rates of flows. While the calculation of fair share in CSFQ requires exponential averaging, the core routers in RFQ only need to perform threshold-based dropping; the routers are thus simple and amenable to hardware implementation. However, the approach applied in the edge routers in their remains complicated. It is important to note that the application of either CSFQ or RFQ requires to the Internet of today extension of the IP packet's header.

We have proposed the basic architecture of an active queue-control scheme, called V-WFQ (Virtual Weighted Fair Queueing), which achieves almost fair

Manuscript received February 1, 2002.
Manuscript revised April 10, 2002.

[†]The authors are with NTT Network Service System Laboratories, NTT Corporation, Musashino-shi, 180-8585 Japan.
[††]The authors are with Electrical Department, Nagaoka University of Technology, Nagaoka-shi, 940-2188 Japan.
[†††]The author is with NTT Network Innovation Laboratories, NTT Corporation, Musashino-shi, 180-8585 Japan.

allocation of bandwidth and minimum bandwidth guarantees in high-speed networks [1], [2]. The word *"Virtual"* here indicates that V-WFQ emulates the function and performance of WFQ without using sophisticated per-flow queueing. In this paper, we present the details of the V-WFQ algorithm and an investigation of its performance through extensive experiments by simulation.

This algorithm's basic process for the fair allocation of bandwidth uses preferential packet dropping to compare the measured input rate of the flow with an estimated fair share of bandwidth. Although V-WFQ is a hardware-efficient FIFO-based algorithm, it is capable of achieving almost ideal fair bandwidth allocation. Unlike CSFQ and RFQ, V-WFQ does not require any extensions to the headers of IP packets and does not need to distinguish between the core and edge routers of the network, so it is easy to apply to today's Internet backbone networks.

2. Proposed Algorithm

2.1 An Overview of V-WFQ

The buffer architecture of a V-WFQ router is shown in Fig. 1. The proposed mechanism: (1) estimates the input rates of the flows at each router in the network, (2) calculates the bandwidth allocated to each flow at each router, and (3) uses FIFO buffering with preferential dropping.

We avoid the need for per-flow buffer management and scheduling by using FIFO buffers with preferential packet discarding on arrival. Thus, as well as providing improved fairness, our proposed mechanism is suited to efficient implementation in high-speed core routers.

A V-WFQ router measures the input rates of the flows it is processing. When each packet arrives, the V-WFQ router calculates a fair bandwidth for the corresponding flow on the basis of the degree of output-link congestion. The router decides whether or not to discard a given packet by comparing the estimated input rate with the calculated result for allocated bandwidth. If the packet is not discarded, it is enqueued in the buffer of the output link.

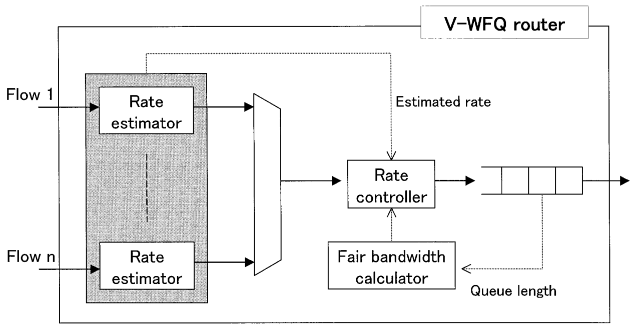

Fig. 1 Buffer architecture of a V-WFQ router.

2.2 The Algorithm of V-WFQ

2.2.1 Estimating Flow Rate

To estimate the packet-arrival rate for a flow, we use the simple 'jumping window' method. This requires only two states for each flow and involves no complex computation to estimate the arrival rate. Let R_i be the estimated rate of flow i and T_w be the window size for using with the method. Let Ct_i be the cumulative packet length of the flow i. At the beginning of the window, Ct_i is set to 0. When a packet of flow i arrives, Ct_i is renewed in the following way:

$$Ct_i = Ct_i + l_i,$$

where l_i is the packet length. At the end of the window, the estimated rate R_i is calculated as

$$R_i = \frac{Ct_i}{T_w}.$$

Ct_i is then reset to 0.

2.2.2 Deciding to Drop or Enqueue a Packet

The V-WFQ router estimates the degree of output-link congestion from the variation in queue length. The following terms are used in calculating the allocated rate:

- N: number of sources,
- $weight_i$: weight of flow i,
- R_i: estimated arrival rate of packets in flow i,
- AR: the fair share for a flow,
- AR_i: the bandwidth allocated to flow i,
- AR_{max}: maximum allocatable bandwidth,
- MGR: per-flow bandwidth achieved in an ideal situation,
- t_{update}: time interval for the updating of AR,
- C: capacity of the output link.

The flowchart and pseudocode of the V-WFQ router algorithm are given in Figs. 2 and 3, respectively. The router updates the fair share AR according to the current congestion status of the output link at the end of every time interval t_{update}. When the average queue length $q_{average}$ reaches the threshold q_{max}, the output link is assumed to be congested and AR should be decreased. If $q_{average}$ is below the threshold q_{min}, the output link is assumed to be underutilized and AR should be increased to raise the degree of link utilization.

When a packet arrives, the router calculates the bandwidth AR_i which is allocated to the flow.

$$AR_i = weight_i \cdot AR$$

Then, the V-WFQ router decides whether or not to discard the arriving packet. We use the random-dropping

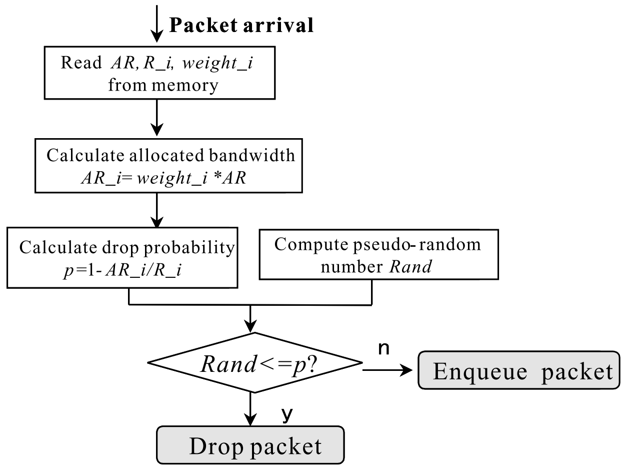
Packet arrival
Read AR, R_i, weight_i from memory
Calculate allocated bandwidth AR_i= weight_i *AR
Calculate drop probability p=1- AR_i/R_i
Compute pseudo-random number Rand
Rand<=p?
n
Enqueue packet
y
Drop packet

Fig. 2 The V-WFQ algorithm.

```
Every time interval t_update
If ($q\_average > q\_max$){
        $AR = MGR$;
}
Else if($q\_average < q\_min$){
        $AR = AR + 0.1AR$;
}
Else{
        $AR = AR/2$;
}
 $AR = \min\{AR, AR\_max\}$;
/* Updating q_average */
 $q\_average = \alpha * q\_now + (1 - \alpha) * q\_average$;


On packet arrival (flowID=i):
/* Calculating the bandwidth allocated AR_i */
 $AR\_i = weight\_i * AR$;
/* Calculating dropping probability p_drop */
If ($R\_i \leqq AR\_i$){
        $p\_drop = 0$;
}
Else if{
        $p\_drop = (R\_i - AR\_i)/R\_i$;
}

If ($p\_drop \geqq Rand$){
        drop packet;
}
Enqueue packet;
```

Fig. 3 Algorithm of packet dropping decision.

approach. The probability of the arriving packet being dropped p_{drop} is given by

$$p_{drop} = \begin{cases} \frac{R_i - AR_i}{R_i} & \text{if } R_i \geq AR_i \\ 0 & \text{if } R_i < AR_i \end{cases}$$

If one ill-behaved flow keeps sending packets at a higher rate than its fair share, the V-WFQ router discards packets of this flow on a probabilistic basis and provides only that bandwidth which corresponds to the calculated fair share for the ill-behaved flow. This mechanism can ensure a fair allocation of bandwidth for all flows that share a given link's resources. Furthermore, by dropping packets long before the buffer becomes full,

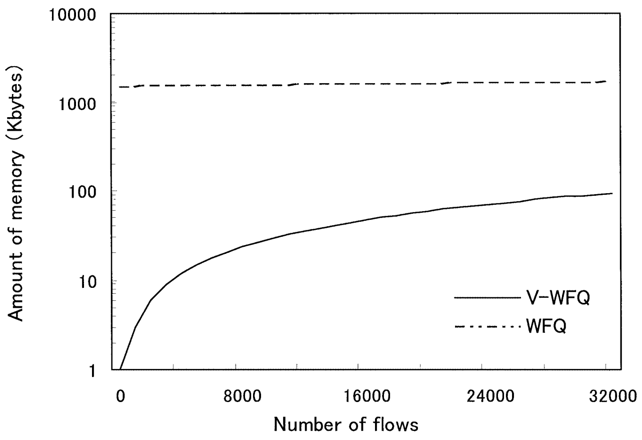

Fig. 4 Comparison of V-WFQ with WFQ in terms of required memory.

this approach avoids the congestion collapse and provides low levels of latency.

2.2.3 Hardware Implementation

Here we compare V-WFQ with WFQ in terms of the cost of hardware implementation. In VLSI, this is directly dependent on the chip area that is required, which in turn is dominated by the requirement for memory [12].

Figure 4 is a comparison of the amount of memory required by V-WFQ and WFQ across a range of numbers of flows. Much less memory is required by V-WFQ than is required by WFQ. For example, V-WFQ only requires about 11 kbytes for 4000 flows, while WFQ requires about 1560 kbytes. V-WFQ is thus a more hardware-efficient algorithm than is WFQ.

However, V-WFQ needs to measure the input rate of each flow and compute a fair bandwidth. The calculation of the fair share consists of simple operations and its computational complexity does not depend on the number of flows. Per-flow traffic measurement is now becoming common in high-speed routers that support line speeds of OC-48 and higher [17]. With today's technology, it is feasible to implement V-WFQ to support several thousands of flows at line speeds of OC-48 and higher.

2.2.4 A New Network Service Based on V-WFQ

We have proposed several new network services that are based on the use of V-WFQ routers [3]. Here we introduce one proposed service. A network which consists of V-WFQ routers is capable of providing a new VPN service as well as a high-quality IP service. Figure 5 shows an isolated-VPN service where the bandwidth is determined on the basis of the tariff concept. The word "isolated" here means that the bandwidth for each VPN is not affected by the number of VPN connections. In

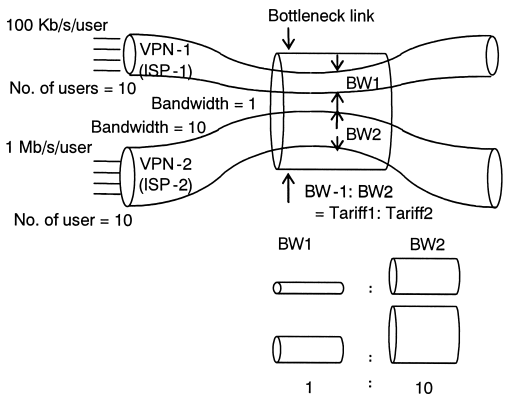

Fig. 5 Example of a new VPN service (BW indicates bandwidth).

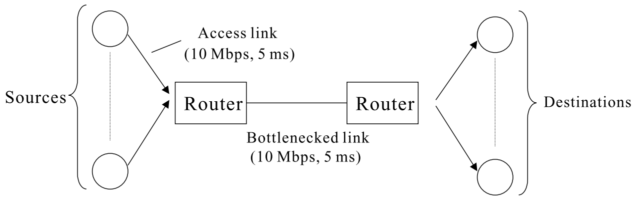

Fig. 6 One-link network model.

other words, the bandwidth for each VPN is provided independently of the other VPNs with which it shares link resources and is determined by the VPN's connection weight alone. Conventional Diff-serv routers can guarantee only a minimum bandwidth. Excess bandwidth is then equally apportioned among the VPNs. In the proposed service, on the other hand, bandwidth in excess of the minimum guaranteed bandwidth is allocated to each of the VPNs according to its weight.

The example shows two ISPs connected to a network and taking advantage of the isolated-VPN service. VPN Service providers can charge each VPN (or ISP) according to the bandwidth with which it is actually provided. For example, the tariff for VPN-1 is \$100/month while that for VPN-2 is \$1000/month; accordingly, VPN-2 has better quality and gets always more bandwidth than VPN-1. Each user can be assured of the required bandwidth even when all go through the same bottleneck link. This service will become the basis of next-generation ISP services and of premium IP services.

3. Performance Evaluation

3.1 Configuration of the Simulation

To investigate the performance of V-WFQ when there is a single congested link in the network, we considered the network configuration shown in Fig. 6. The congested link is between two routers. The link, which has a capacity of 10 Mbps, is shared by m TCP flows and n UDP flows. Each end host is connected to the router via a 10-Mbps link and one end host persistently transmits packets to the other end host. We compared the performance of V-WFQ with Random Early Discard (RED) [13], Flow Random Early Discard (FRED) [7], Deficit Round Robin (DRR) [14], CSFQ, and DropTail.

RED monitors the average queue length and starts to drop arriving packets on a probabilistic basis when the average queue length exceeds a certain threshold. To make a router start to drop packets before its buffer becomes full, RED gives an early congestion indication to each flow and avoids congestion collapse. However, RED does not provide fairness among competing flows. FRED brings fairness of a bandwidth allocation to RED by maintaining a per-flow state. FRED drops packets from flows that have had many packets dropped in the past and from flows that have longer queues than the average queue length.

DRR is a well-known implementation of WFQ that has a computational complexity $O(1)$. The buffer management scheme of DRR requires a sophisticated per-flow queueing mechanism, while the other algorithms applied in our simulation only require simple FIFO-buffer mechanisms. DRR achieves a higher degree of fairness than the other algorithms, so we used DRR as the benchmark for the fair allocation of bandwidth.

All simulations were performed in ns-2 [15]. The simulation code for the DRR, RED, FRED, and CSFQ algorithms is supplied with the ns-2 package. In all simulations, unless otherwise stated, we used the following assumptions and parameters.

- All TCP sources were persistent TCPs and started at the same time. For a TCP flow, the maximum segment size (MSS) was 1000 bytes and the version of TCP was Reno. The TCP hosts transmitted files of infinite size by FTP to the corresponding end hosts.
- The UDP hosts sent packets at a constant bit rate (CBR) of k kbps, where k is a variable, and all started sending at the same time. The packet size of the UDP flows was set to 500 bytes.
- The output buffer size was 128 KB.
- All flows had the same weight values.
- The following parameters were used with V-WFQ. The jumping window size was 50 ms. q_{max} and q_{min} were 128 KB and 8 KB, respectively. Fair sharing MGR was 625 kbps and AR_{max} was 10 Mbps. α was 0.05.

3.2 Indices of Performance

We evaluated the performance of our algorithms in terms of fairness and efficiency. Let N be the number of TCP sources, C be the capacity of the bottleneck link, x_i be the measured throughput of the i-th TCP source, and z_i be the ideal throughput of the i-th TCP source. We define efficiency as

$$\textit{efficiency} = \frac{\sum_{i=1}^{N} x_i}{C}.$$

We measured the fairness of algorithms in terms of the *fairness index* (FI) [16], which is defined as

$$\textit{fairness index} = \frac{\left(\sum_{i=1}^{N} \frac{z_i}{x_i}\right)^2}{N \times \sum_{i=1} \left(\frac{z_i}{x_i}\right)^2}.$$

3.3 Results of Simulation

3.3.1 Fair Allocation for UDP Flows

In the first experiment, we evaluated the performance of V-WFQ when all hosts were transmitting an infinite amount of data according to the UDP. Each of the 16 UDP sources sent packets at $i \times \frac{10}{16}$ kbps, where $i\,(1, \ldots, 16)$ is the flow ID. Each UDP source constantly had data to transmit so the backbone link was always severely congested. Under max-min fairness [16], each flow should have achieved an average throughput of 626 kbps. Figure 7 shows the average throughput achieved by the 16 UDP sources sharing the bottleneck link which was configured with V-WFQ, DRR, CSFQ, FRED, RED or DropTail. Table 1 shows the index of fairness and the level of efficiency for each algorithm used in this simulation. The throughput achieved by RED or DropTail was almost exactly in proportion to the input rate of each flow, while the throughput achieved by V-WFQ, DRR, or CSFQ was nearly constant at around 626 kbps. These results show that RED and DropTail completely failed to allocate fair amounts of bandwidth to the flows during periods of congestion while DRR and CSFQ achieved almost ideal fairness in sharing bandwidth among competing flows. V-WFQ had the same performance as DRR and much better performance than FRED.

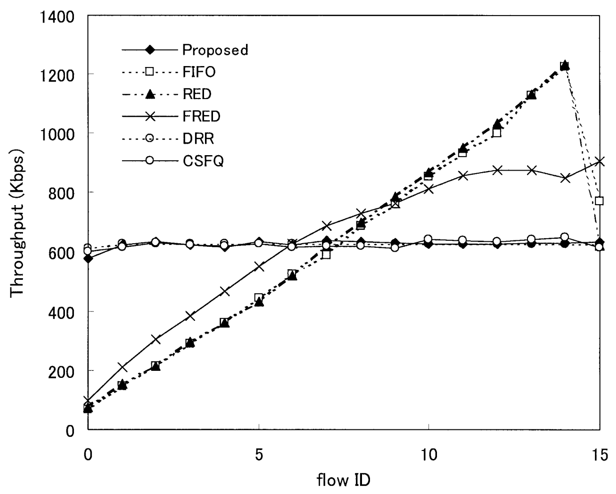

Fig. 7 Throughput for 16 UDP flows sharing a 10-Mbps bottleneck link. Each UDP flow i sends packets at i times its fair allocation (626 kbps).

3.3.2 A Single Misbehaving Flow

In the second experiment, we examined the levels of protection provided by the various algorithms for well-behaved flows in the presence of a single misbehaving flow. In this simulation, we assumed that 15 TCP flows and 1 UDP flow shared a 10-Mbps bottleneck link. The UDP source sent packets at the constant bit rate of 10 Mbps during the 30 seconds of simulation. Figure 8 shows the normalized throughput, which is defined as the throughput/fair-share ratio. Note that the normalized throughput for each flow should equal 1.0 in the optimal situation. Table 2 shows the fairness index and level of efficiency achieved by each algorithm used in this simulation.

Table 1 Comparison of fairness indices and efficiency.

	Efficiency	Fairness
V-WFQ	0.999	0.999
DRR	0.999	0.999
CSFQ	0.999	0.999
FRED	0.999	0.859
RED	0.999	0.766
DropTail	0.999	0.768

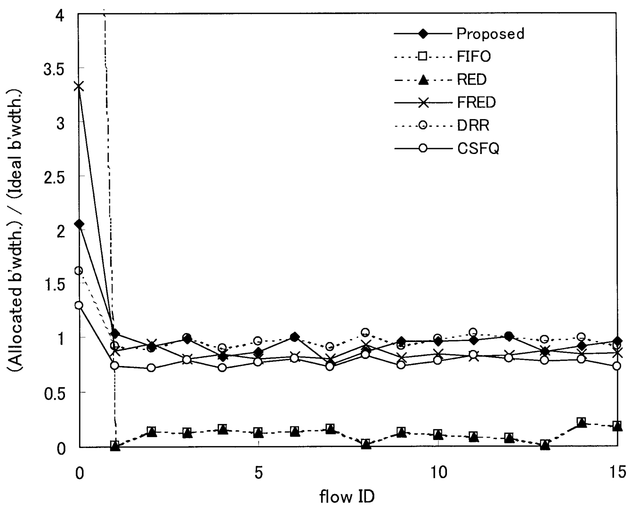

Fig. 8 Throughput for 1 UDP flow and 15 TCP flows sharing a 10-Mbps bottleneck link. The UDP flow i sends packets at 10 Mbps.

Table 2 Indices of efficiency and fairness for a configuration with a single misbehaving UDP flow and 15 TCP flows.

	Efficiency	Fairness
V-WFQ	0.995	0.926
DRR	1.00	0.973
CSFQ	0.996	0.952
FRED	1.00	0.735
RED	1.00	0.0778
DropTail	1.00	0.0778

Under FIFO and RED, the misbehaving UDP flow got more than 8 Mbps, while each of the other TCP flows received less than 30 percent of fair share. Under V-WFQ, the UDP flow got about twice the fair bandwidth, but most of the TCP flows got more than 85 percent of the fair bandwidth. CSFQ shows similar tendencies. Under DRR, most of the flows got almost the ideal bandwidth. These results show that RED and DropTail failed to provide fair share to the well-behaved TCP flows while DRR, CSFQ, and V-WFQ protected the TCP flows against the over-transmitting UDP flow and achieved almost ideal fairness. On fairness index, CSFQ slightly outperformed V-WFQ. Both CSFQ and V-WFQ are FIFO-based algorithms and require per-flow traffic measurement. In V-WFQ, the simple jumping-window method is used to measure per-flow traffic, while CSFQ requires complex exponential averaging at a high-speed core router. The difference in performance is traded-off against the cost of implementation.

3.3.3 Flows with Different Weights

Next, we investigated the performance of V-WFQ for flows with different weights. We compared the performance of V-WFQ with DRR in terms of throughput achieved for each flow and of queue dynamics. Here, the number of source-destination pairs was 5. The weight $weight_i$ is given by

$$weight_1 : weight_2 : \cdots : weight_5 = 1 : 2 : \cdots : 5.$$

In this simulation, the sources were all TCP sources. During the 30-s simulation, each TCP source had an infinite amount of data to transmit so the backbone link was always severely congested. The ideal result is the allocation of bandwidth to each flow in proportion to its weight.

Figure 9 shows the average throughput achieved for the 5 TCP sources over a 30-s interval. Under RED and FIFO, the allocation of shared bandwidth to each flow was independent of its weight. The throughput achieved by V-WFQ or DRR was proportional to its weight. RED and FIFO failed to ensure fairness in terms of throughput, while V-WFQ and DRR achieved almost ideal fairness.

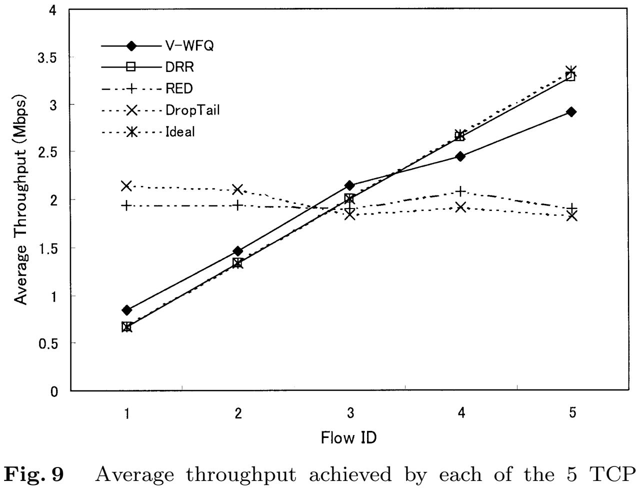

Fig. 9 Average throughput achieved by each of the 5 TCP flows with different weights.

Figure 10 compares the queue dynamics for V-WFQ and DRR. The queue length of DRR is the sum of the length of the separate per-flow queues. The queue length for V-WFQ fluctuated greatly between 0 kbytes and 60 kbytes, while the length for DRR tended to stabilize at a higher level of buffer occupancy, i.e., around 90 kbytes. V-WFQ estimated its fair allocations for each flow from the queue length, hence the rate control of a V-WFQ router ceases to operate the queue length is not becoming longer. The average queue length was much shorter for V-WFQ than for DRR. V-WFQ, like RED, starts to drop packets before the buffer becomes full, so it gives an early congestion indication to each flow and hence can provide much lower latency than DRR.

4. Concluding Remarks

We have proposed a FIFO-based mechanism called V-WFQ that achieves fairness in throughput and have investigated its performance in terms of both fairness and link utilization through extensive simulations. In this investigation, we compared the performance of V-WFQ with DRR and other algorithms. DRR requires a sophisticated per-flow queueing mechanism, while V-WFQ only requires simple FIFO-buffer mechanisms. DRR achieved a higher degree of fairness than the other algorithms, so we used DRR as the benchmark for the fair allocation of bandwidth. The results of simulation show that V-WFQ achieves almost ideal fairness under various simulated conditions. In particular, we should note that it showed almost the same levels of performance as DRR and achieved a much higher degree of fairness than FRED, which, like V-WFQ, is a preferential dropping algorithm.

It is also possible to use the V-WFQ architecture to provide minimum bandwidth guarantees for flows. One example of a new application which V-WFQ allows is

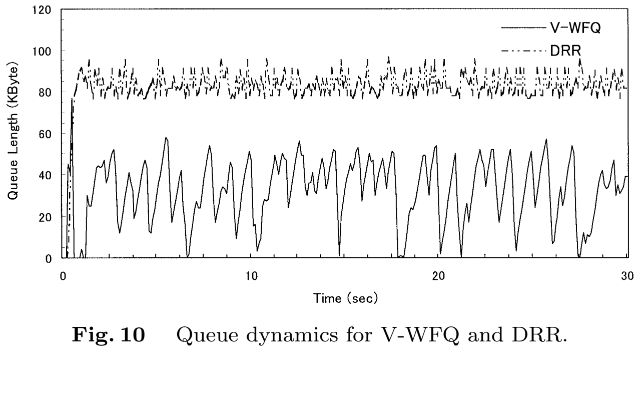

Fig. 10 Queue dynamics for V-WFQ and DRR.

an IP VPN backbone network that provides minimum bandwidth guarantees to each of multiple VPNs.

References

[1] T. Miyamura, T. Kurimoto, K. Nakagawa, P. Dhananjaya, M. Aoki, and N. Yamanaka, "A new queueing mechanism for achieving fair bandwidth allocation in high-speed networks," Proc. APCC 2001, vol.E84-B, pp.473–476, 2001.
[2] T. Miyamura, T. Kurimoto, K. Nakagawa, P. Dhananjaya, M. Aoki, and N. Yamanaka, "Active queue control scheme for achieving approximately fair bandwidth allocation," Proc. IEEE ICC, vol.2, pp.1269–1273, 2002.
[3] N. Yamanaka, T. Kurimoto, T. Miyamura, and M. Aoki, "MSN Type-X: Next generation Internet backbone switch/router architecture," Proc. IEEE ICC, vol.4, pp.2085–2099, 2002.
[4] A. Parekh and G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: The single-node case," IEEE/ACM Trans. Networking, vol.1, no.3, pp.344–357, 1993.
[5] S.J. Golestani, "A self-clocked fair queueing scheme for broadband applications," Proc. IEEE INFOCOM, vol.2, pp.636–645, 1994.
[6] S. Suri and G. Varghese, "Leap forward virtual clock: A new fair queueing scheme with guaranteed delays and throughput fairness," Proc. IEEE INFOCOM, vol.3, pp.557–565, 1997.
[7] M. Schreedhar and G. Varghese, "Efficient fair queueing using deficit round robin," Proc. SIGCOMM'95, Sept. 1995.
[8] M. Katevenis, S. Sidiropoulos, and C. Courcoubetis, "Weighted round-robin cell multiplexing in a general-purpose ATM switch chip," IEEE J. Sel. Areas Commun., vol.9, no.8, pp.1265–1279, 1991.
[9] I. Stoica, S. Shenker, and H. Zhang, "*Core*-stateless fair queueing: Achieving approximately fair bandwidth allocations in high-speed networks," Proc. ACM SIGCOMM'98, pp.118–130, Sept. 1998.
[10] Z. Cao, Z. Wang, and E. Zegura, "Rainbow fair queueing: Fair bandwidth sharing without per-flow state," Proc. IEEE INFOCOM 2000, vol.2, pp.922–931, March 2000.
[11] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, L. Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski, and L. Zhang, "Recommendations on queue and congestion avoidance in the Internet," IETF, RFC2309, April 1998.
[12] M. Grossglauser and S. Keshav, "On CBR service," Proc. IEEE INFOCOM, pp.129–137, 1996.
[13] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," IEEE/ACM Trans. Networking, vol.1, no.4, pp.397–413, 1993.
[14] D. Lin and R. Morris, "Dynamics of random early detection," Proc. SIGCOMM'97, pp.127–137, Oct. 1997.
[15] UCB/LBNL/VINT, "Network Simulator - NS (Version 2)," http://www-mash.cs.berkley.edu/ns/
[16] R. Jain, A. Durresi, and G. Babic, "Throughput fairness index: An explanation," ATM Forum, 99-0045, 1999.
[17] Cisco Systems Inc., "Cisco Documentation," http://www.cisco.com /univercd/home/home.htm

Takashi Miyamura received the B.S. and M.S. degrees from Osaka University, Osaka, Japan, in 1997 and 1999, respectively. In 1999, he joined NTT (Nippon Telegraph and Telephone Corp.) Network Service Systems Laboratories, where he was engaged in research and development of a high-speed IP switching router. He is now researching future photonic IP networks and an optical switching system. He received Paper Awards from the 7th Asia-Pacific Conference on Communications in 2001. He is a member of the Operations Research Society of Japan.

Takashi Kurimoto graduated from the Tokyo Institute of Technology, Japan, where he received B.E. and M.E. degrees in applied physics 1992 and 1994, respectively. In 1994, he joined Nippon Telegraph and Telephone Corporation's (NTT's) Network Service Systems Laboratories, Tokyo Japan. He has been engaged in researching the switching technology for high-speed computer networks and multimedia communication networks.

Kenji Nakagawa received the B.S., M.S. and D.S. degrees from Tokyo Institute of Technology, Tokyo, Japan, in 1980, 1982 and 1986, respectively. In 1985, he joined NTT (Nippon Telegraph and Telephone Corp.). Since 1992, he has been an associate professor of Dept. of Electrical Engineering, Nagaoka University of Technology. He is an associate editor of the IEICE Transactions on Communications. His research interests include queueing theory, performance evaluation of networks, and geometric theory of statics. Dr. Nakagawa is a member of the IEEE, SITA, and Mathematical Society of Japan.

Prasad Dhananjaya received the B.E., M.E. degrees in Electrical Engineering from Nagaoka University of Technology, Niigata, Japan, in 2000 and 2002, respectively. He joined Fujitsu Primesoftware Technologies Ltd in 2002. His research interests include the performance evaluation of TCP/IP networks.

Michihiro Aoki received the B.E. and M.E. degrees in electronics engineering from Chiba University, Chiba, Japan, in 1981 and 1983, respectively. In 1983, he joined the Electrical Communication Laboratory, Nippon Telegraph and Telephone Public Corporation, where he has been engaged in the research and development of high reliable and high performance switching systems and high performance IP router. He is currently a senior research engineer of the NTT Network Service Systems Laboratories. He received the Asia Pacific Conference on Communications Paper award in 2001. He is a member of the Information Processing Society of Japan.

Naoaki Yamanaka graduated from Keio University, Japan where he received B.E., M.E. and Ph.D. degrees in engineering in 1981, 1983 and 1991, respectively. In 1983 he joined Nippon Telegraph and Telephone Corporation's (NTT's) Communication Switching Laboratories, Tokyo Japan, where he was engaged in research and development of a high-speed switching system and high-speed switching technologies for Broadband ISDN services. Since 1994, he has been active in the development of ATM base backbone network and system including Tb/s electrical/optical backbone switching as NTT's Distinguished Technical Member. He is now researching future optical IP network, and optical MPLS router system. He is currently a senior research engineer, supervisor, and research group leader in Network Innovation Laboratories at NTT. He has published over 112 peer-reviewed journal and transaction articles, written 82 international conference papers, and been awarded 174 patents including 17 international patents. Dr. Yamanaka received Best of Conference Awards from the 40th, 44th, and 48th IEEE Electronic Components and Technology Conference in 1990, 1994 and 1998, TELECOM System Technology Prize from the Telecommunications Advancement Foundation in 1994, IEEE CPMT Transactions Part B: Best Transactions Paper Award in 1996 and IEICE Transaction Paper award in 1999. Dr. Yamanaka is Technical Editor of IEEE Communication Magazine, Broadband Network Area Editor of IEEE Communication Surveys, Former Editor of IEICE Transaction, TAC Chair of Asia Pacific Board at IEEE Communications Society as well as Board member of IEEE CPMT Society. Dr. Yamanaka is an IEEE Fellow.