SF-FC: A Neighbor-State Based Flow Control with Soft Fairness*

Yosuke Tamura †, Yoshito Tobe †, and Hideyuki Tokuda †II

† Graduate School of Media and Governance,
II Faculty of Environmental Information,
Keio University, 5322 Endo Fujisawa, Kanagawa, 252-8520, JAPAN
{*tamura, tobe, hxt*}*@sfc.keio.ac.jp*

Abstract

Many audio and video stream applications in the Internet require good network performance but not perfect reliability. Such applications should employ congestion control and should not shut down the TCP flows by restricting buffering too much.

However, the Internet is not a small local community, and it is impossible to rely on all developers to incorporate end-to-end congestion control in their Internet applications. The network should take part in controlling its own resource utilization.

Most research for fair queuing tend to focus on reducing processing overhead and achieving exact fairness for each flow. In this paper we claim that perfect fair queuing is not effective for end users in the current complicated network environment.

We propose Soft Fair Flow Control(SF-FC) which is a light weight router mechanism that employs simple FIFO queuing, and shows high performance than other fair queuing. SF-FC provides fair network resource allocation to non-adaptive and adaptive flows. Each SF-FC router informs the upstream SF-FC router of its link state. Based on the received state, SF-FC dynamically sets the packet discard rate of the non-adaptive flows.

1 Introduction

In the Internet where various network applications share the network resources, flow control needs to be combined with congestion control. In recent years congestion control relied only on the mechanisms provided by Transmission Control Protocol(TCP)[2]. TCP congestion control becomes efficient only when all the flow cooperate, and fairness is achieved when all the flows use the same congestion control algorithm.

On the other hand, in the Internet several real-time streaming applications run over User Datagram Protocol(UDP)[1] using an individual flow control for improving their performance. Like Integrated Service/Resource reSerVation Protocol(RSVP)[12] and Differentiated Service[4], it is important to provide Quality of Service(QoS) guarantees to these applications. However, network resources are limited and it is impossible to grant the QoS demands of all applications. Therefore, we should not guarantee QoS without considering fairness.

Unfair sharing among adaptive flows of different segment size packets and Round Trip Time(RTT) are not much of a problem compared to unfair sharing of adaptive and non-adaptive flows. By controlling non-adaptive flows appropriately, the Internet will be a disciplined environment.

Like TCP-friendly[5] schemes, it is important for non-TCP flows to have congestion control to keep the fairness with TCP flows. However, flow control inside the network is also necessary because end hosts may not always have good intentions. For that reason, it is favorable to control flows at the router and design the network from a network administrator standpoint.

Various types of fair queuing, such as Deficit Round Robin(DRR)[10] have been proposed. However, they cannot be easily deployed in a backbone environment with thousands of flows, as their complexity increases with the number of flows. Since fair scheduling is done toward output links, whether efficient control has been made to end applications is hard to estimate, especially in a complicated network environment where one may pass multiple routers.

In this paper, we propose Soft Fair Flow Control (SF-FC) from the network administrator standpoint. *Soft Fair* means that it does not achieve strict per-flow

*This research is conducted under the fund of Information-technology Promotion Agency, Japan(IPA).

fairness like DRR. In SF-FC, the non-adaptive flow is controlled based on the flow state from the downstream router, and it solves the wasted allocation problem in current fair queuing. SF-FC also achieves a highly efficient fair bandwidth allocation, so that the bandwidth is not monopolized by non-adaptive flows.

The paper is organized in the following order. Section 2 of the paper covers the problem of current queuing models. In sections 3 and 4, we propose our scheme and the implementation. Section 5 compares SF-FC with other queuing models. Sections 6 and 7 include discussion and future works, followed by a conclusion.

2 Problems in Current Fair Queuing

Most of the fair queuing specify a mechanism within a single router, and do not define any cooperation between routers. If all flows pass one router and reach the final destination in one hop, then fair queuing can provide aimed performance to end-users.

However, in many cases flows pass multiple routers with different policy and network bandwidth. Therefore, there is no guarantee that the allocated bandwidth in the upstream router will be allocated again in the downstream router.

We assume that a flow arrives at its final destination through n fair queuing routers. The number of flows that share the output link of the i th router is represented as fn_i. The bandwidth of the output link is bw_i.The following equation shows the gross bandwidth wastefully allocated to the flow, where x^* denotes $max(x, 0)$.

$$wasted\ allocation = \sum_{i=1}^{n-1} \left(\frac{bw_{i+1}}{fn_{i+1}} - \frac{bw_i}{fn_i}\right)^*$$

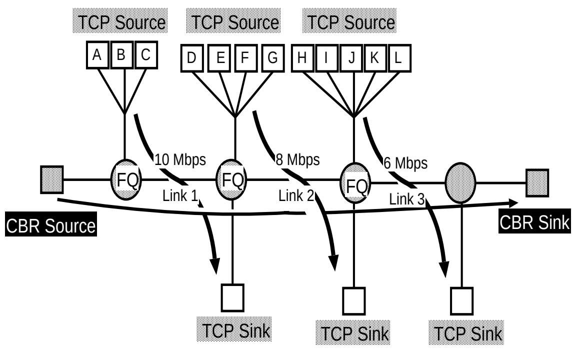

Figure 1. Test Environment

Here, we show an example. We assume that a Constant Bit Rate(CBR) flow passes three fair queuing routers as seen in Figure 1. In Link 1, 10-Mbps bandwidth is shared by three TCP flows (flow A, B, C) and one CBR flow. In Link 2, 8-Mbps bandwidth is shared by four TCP flows (flow D, E, F, G) and one CBR flow. In Link 3, 6-Mbps bandwidth is shared by three TCP flows (flow H, I, J, K, L) and one CBR flow. Figure 2 shows the bandwidth utilization of the flows in each link. The CBR flow is utilized with different amounts of bandwidth at each link. However, at Link 3, the fair share bandwidth and final throughput of the CBR flow are 1 Mbps, which means that there is wasted allocation in Link 1 and Link 2.

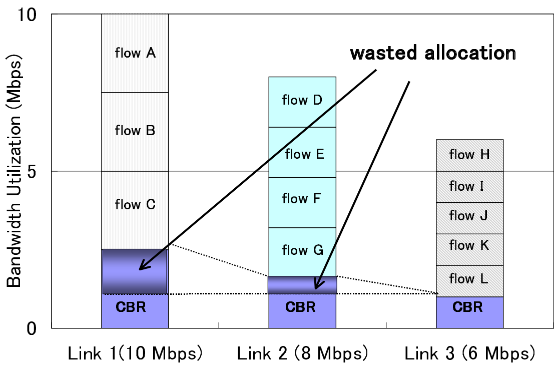

Figure 2. Wasted Allocation

3 Neighbor-state Based Soft Fair Flow Control

In this section, we describe the design of Soft Fair Flow Control (SF-FC). SF-FC is designed based on the Neighbor-state Based Queuing(NBQ)[11]. SF-FC expands NBQ's flow control to achieve higher performance and fairness between non-adaptive and adaptive flows.

SF-FC consists of three principal elements.

- SF-FC Manager that exchanges the status of non-adaptive flow with neighboring routers.
- SF-FC Estimator that identifies non-adaptive and adaptive flows and estimates the status.
- SF-FC Discarder that controls sending rate by discarding the packet of non-adaptive flows.

3.1 Estimate and Maintain Flow Status

Table 1 shows the variables that SF-FC maintains. Non-adaptive and adaptive flows are identified from the protocol field in the IP header. We use τ to calculate the number of active flows.

Two valuables, ρ and υ, are calculated by the condition of the flow received from the downstream router and the measurement result from other variables. Detailed calculation of ρ and υ are presented in the next section.

3.2 Negotiation between Neighboring Routers

SF-FC Manager of the downstream router monitors packet arrival quantity $\tilde{\lambda}$ and packet departure quantity $\tilde{\mu}$ at regular intervals. SF-FC Manager informs their link state to the upstream router only when the output link is congested and there is a buffer overflow that causes packets to be dropped.

The mechanism of searching the neighboring router is shown in Figure 3. First the SF-FC Manager checks whether it has an entry of incoming flow. If an entry exists, the SF-FC Manager will use its IP address. If an entry does not exist, the SF-FC Manager will broadcast the Neighboring Discovery Packet(NDP).

If there is no response from the neighboring SF-FC Managers, the NDP is resent toward the source host of the flow. Since NDP has a special protocol identifier, if it passes a router that supports SF-FC, it will be received there without going to the final address. The router that receives the NDP answers back to the router that sent it.

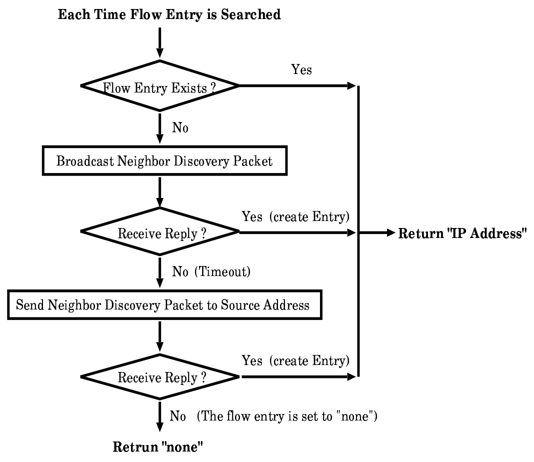
Each Time Flow Entry is Searched
Flow Entry Exists ?
Yes
No
Broadcast Neighbor Discovery Packet
Receive Reply ?
Yes (create Entry)
Return "IP Address"
No (Timeout)
Send Neighbor Discovery Packet to Source Address
Receive Reply ?
Yes (create Entry)
No (The flow entry is set to "none")
Retrun "none"

Figure 3. Searching Neighbor Routers

4 Flow Control Mechanism

Basic Discard Rate (BDR) and Fair Bandwidth (F-B) are keywords in the flow control of SF-FC. The final packet discard rate υ is calculated using BDR and FB.

- **BDR** is the packet discard rate of non-adaptive flows calculated so that it does not exceed the packet departure quantity in the downstream router.
- **FB** is the bandwidth that is allocated to non-adaptive flows in consideration of the fairness with adaptive flows.

4.1 Calculation for Basic Discard Rate

Based on the notification from the downstream router, SF-FC calculates the BDR of each non-adaptive flow. We explain the calculation method of BDR in Figure 4.

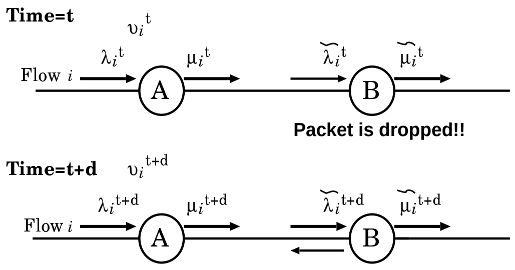

Figure 4. Basic Discard Rate Estimation

Increasing BDR

We assume $\upsilon_i{}^t$ as the packet discard rate of flow i at time t in router A. The relationship between packet arrival quantity and packet departure quantity is shown in Eq. (1). When a packet drop occurs due to network congestion in router B at time t, router A increases $\upsilon_i{}^t$ to reduce the packet departure quantity.

$$\lambda_i{}^t(1-\upsilon_i{}^t) = \mu_i{}^t \tag{1}$$

Router B notifies router A of $\tilde{\lambda}_i^t$ and $\tilde{\mu}_i^t$. We assume that the notified packet arrived at router A at time $t+d$. Router A can calculate the packet discard rate $\upsilon_i{}^{t+d}$ without packet drop in router B at time $t+d$ as follows:

$$\begin{aligned}\upsilon_i{}^{t+d} &= 1 - \frac{\tilde{\mu}_i{}^t}{\tilde{\lambda}_i{}^t}\frac{\lambda_i{}^t}{\lambda_i{}^{t+d}}(1-\upsilon_i{}^t) \\ &= 1 - \frac{\tilde{\mu}_i{}^t}{\tilde{\lambda}_i{}^t}\frac{\mu_i{}^t}{\lambda_i{}^{t+d}} \end{aligned} \tag{2}$$

Table 1. Estimate Variables

Symbol	Estimate Variable	Type	Flow
γ	Arrival Packet Quantity	Unique	Adaptive Flows
τ	Last Packet Arrival Time	Per-Flow	Adaptive & Non-Adaptive Flows
λ	Arrival Packet Quantity	Per-Flow	Non-Adaptive Flows
μ	Departure Packet Quantity	Per-Flow	Non-Adaptive Flows
ρ	Arrival Packet Regulation	Unique†	Adaptive Flows
υ	Packet Discard Rate	Per-Flow†	Non-Adaptive Flows

Since router A does not know the exact value of d, it cannot recognize ${\mu_i}^t$ when a packet is dropped in router B. In order to avoid this problem, we take the regular number from the history of μ_i in router A, and set its mean value as $\bar{\mu}_i^t$.

The basic discard rate (BDR_i) is updated at regular interval N. It is desirable to set a larger value for N than the network delay to the downstream router. BDR_i is increased by using exponential averaging as shown in Eq.(3).

$$BDR_i = \alpha {\upsilon_i}^t + (1-\alpha){\upsilon_i}^{t+N} \qquad (0 < \alpha < 1) \qquad (3)$$

Decreasing BDR

As describe in the previous section, SF-FC Manager informs their link state to the upstream router only when there is a buffer overflow that causes packets to be dropped. When SF-FC Manager does not receive their link state information packet from downstream routers at a regular interval, the sending quantity is increased by a fixed quantity(I) using Eq. (4).

When the network traffic is reduced and the packet drop in the downstream router $(1 - \frac{\bar{\mu_i}^t}{\bar{\lambda_i}^t})$ becomes below a regular value, the upstream router also reduces the packet discard rate with Eq. (4).

$$BDR_i = {\upsilon_i}^t - \frac{I}{{\lambda_i}^t} \qquad (4)$$

4.2 Calculation for Fair Bandwidth

Non-adaptive flows can be controlled efficiently when an upstream router uses BDR. However, in the Internet, there are adaptive flows that conduct flow control at end hosts. Therefore, SF-FC needs to consider the coexistence of non-adaptive and adaptive flows.

FB is the amount of bandwidth that non-adaptive flows can utilize and is calculated in advance to avoid over-utilization of bandwidth. The final packet discard rate υ is calculated based on the FB and BDR.

The number of active non-adaptive flow n_n and the number of active adaptive flow n_a are counted by using τ in SF-FC. The theorical value of the bandwidth allocated to non-adaptive flow W is calculated with Eq.(5) using the output link bandwidth B. The bandwidth that non-adaptive flows are actually using $\hat{W}$ is calculated with Eq.(5) by using packet arrival quantity λ_i and packet discard rate υ_i.

$$W = B\frac{n_n}{n_n + n_a}, \qquad \hat{W} = \sum_{i=1}^{n_n} \mu_i \qquad (5)$$

If there is a bottleneck in the downstream link, there is a possibility that the adaptive flow cannot use all the allocated bandwidth. In this case, it is better for SF-FC to allocate the unused bandwidth to non-adaptive flows.

The change rate ρ in the arrival packet quantity of adaptive flow is being monitored by using Eq.(6). When the change rate is stabilized, all the available bandwidth is allocated to non-adaptive flows. Since adaptive flows always yield bandwidth to non-adaptive flows, it is impossible for adaptive flows to invade or unfairly occupy bandwidth. Therefore, SF-FC makes sure that non-adaptive flows do not exceed the fair share limit by pushing away adaptive flows. FB is calculated on the basis of Eq.(7).

$$\rho^t = \beta\rho^{t-1} + (1-\beta)\frac{\gamma^{t-1}}{\gamma^t} \qquad (0 < \beta < 1) \qquad (6)$$

$$FB = \begin{cases} \hat{W} & if \quad \hat{W} < W \\ B - \gamma & else\,if \quad 0.9 < \rho < 1.1 \\ W & otherwise \end{cases} \qquad (7)$$

Based on max-min fair allocation in Figure 5, the bandwidth allocated to each flow is calculated from FB and BDR. The final packet discard rate υ is calculated from the allocated bandwidth and the arrival packet quantity.

```
max_min_estimate (buffer, count) {
  allocation = buffer / ( $n_a$ + $n_n$ - count );
  for ( i = 0; i < $n_n$ ; i ++ ) {
      if ( $\lambda^i$ != $\delta^i$ )
          $\delta^i$    += allocation;
      if ( $\lambda^i$ <= $\delta^i$ ) {
          rest  += $\delta^i$ - $\lambda^i$ ;
          $\delta^i$    =  $\lambda^i$ ;
          count ++;
      }
  }
  if (( rest > 0 ) && (count < $n_n$ ))
       max_min_estimate (rest, count);
}
```

Figure 5. max_min_estimate function

The max_min_estimate function allocates the FB to each non-adaptive flow. We denote δ_i as the allocated fair bandwidth to non-adaptive flow i. v_i is calculated by 1 - $\frac{\delta_i}{\lambda_i}$

5 Evaluation

In this section, we evaluate the SF-FC performance by simulation. All simulations were performed using the *ns-2*[9] simulator. To evaluate the performance of the SF-FC, we compared SF-FC with FIFO and DRR[10]. We repeat the same experiment 100 times for 30 seconds and take the mean value as data. Each link has a delay of 1 ms, and a buffer of 64 KB. The TCP implementation is a BSD compatible two-way TCP based on TCP-Reno. The packet payload size is 1000 byte. Delayed ACK is enabled.

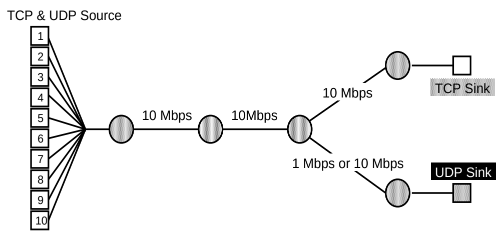

Figure 6. Simulation Network Topology

5.1 Performance of Fairness

To compare the performance of fairness, we simulated configuration shown in Figure 6. The bandwidth of the link that is connected to UDP sink is set to 10 Mbps. There are a total of 10 flows that share two links. In all experiments, the flow of UDPN is CBR of the transfer rates of $N \times 2$ Mbps. Figure 7 shows the throughput of each flow where all flows are UDP. SF-FC can allocates fair bandwidth to each CBR that differs from DRR without requiring multiple queues and complicated scheduling.

Figure 8 shows the throughput of one TCP flow and nine UDP flows. Figure 9 shows the throughput of five TCP and UDP flows. Unlike FIFO, SF-FC can allocate a fair bandwidth to the TCP flows. In FIFO, non-adaptive flows like UDP degrade TCP performance substantially. For TCP such phenomenon becomes a serious problem. In FIFO, depending on the input rate of UDP, unfair sharing occurs among UDP flows.

In these simulation results, one can see that the performance of SF-FC and DRR are similar. However, since SF-FC processes multiple adaptive flows as one group, fairness cannot be achieved among adaptive flows with different segment size or RTT. Regarding the performance of fairness, DRR shows excellent performance.

5.2 Performance of Link Sharing

The simulation environment in the previous section assumed an ideal environment for fair queuing routers. All the links on the network had the same link bandwidth and was shared by the same number of flows. However, in reality, such an environment hardly exists and usually there is a bottleneck that occurs from network congestion or narrow bandwidth. Here we simulated the SF-FC performance where there is a bottleneck at the downstream link. Details are shown in Figure 6 where the bandwidth of the link connected to the UDP sink is set to 1 Mbps.

The simulation result is shown in Figure 10. While DRR and SF-FC both have the same amount of UDP throughput, SF-FC shows higher performance in TCP throughput. This is due to the fact that the downstream is a bottleneck and SF-FC avoids bandwidth allocation to wasteful packets in upstream router.

Also, we simulated the performance of SF-FC in a wide area network environment shown in Figure 11. 10-Mbps UDP flow passes ten routers. The UDP sink is connected to a bottleneck link of 1 Mbps. Each of the five TCP flows share nine links connected to nine routers. The TCP sources connected to GN, transmit data to the TCP sink connected to G$(N+1)$. The mean value for each of the forty-five TCP transfer rates are shown in Figure 12. In FIFO the bandwidth is mostly occupied by UDP and TCP has a low transfer rate. As can be seen, SF-FC achieves the highest performance.

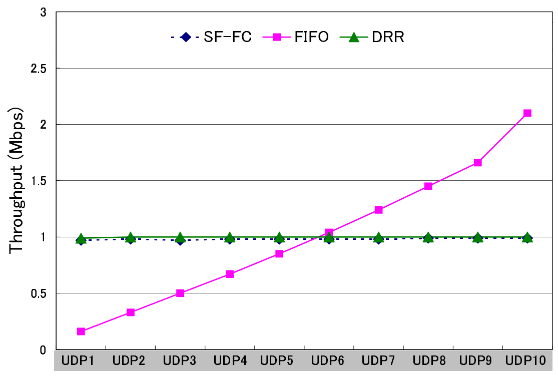

Figure 7. Performance of Each Flow (10 CBR flows)

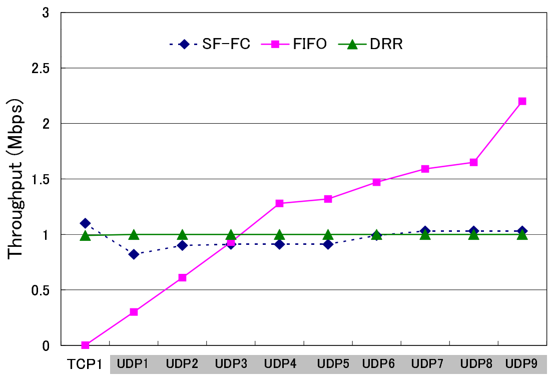

Figure 8. Performance of Each Flow (1 TCP flow and 9 CBR flows)

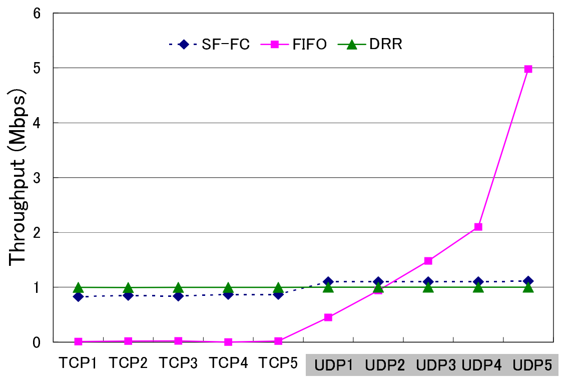

Figure 9. Performance of Each Flow (5 TCP flows and 5 CBR flows)

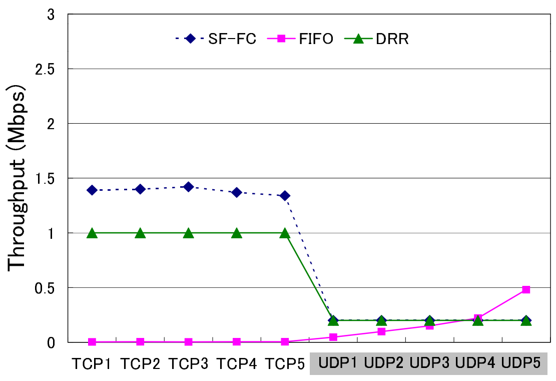

Figure 10. Simulation Result

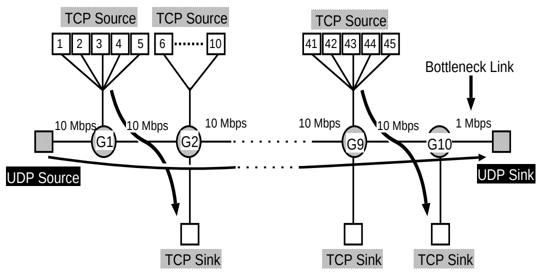

Figure 11. Simulation Network Topology

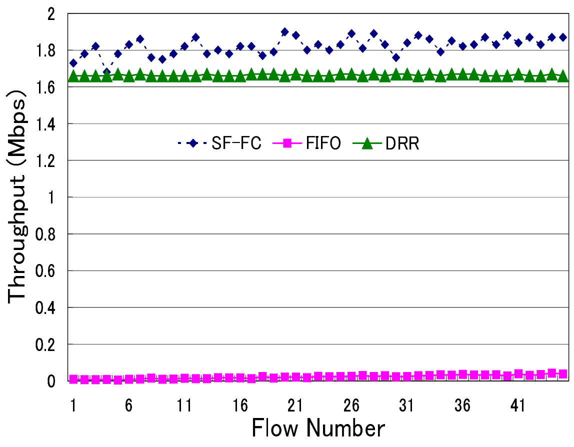

Figure 12. Simulation Result

5.3 Discussion on the Simulation Results

Even if we introduce a complete fair queuing like DRR, under a complicated network environment it will not be effective in performance improvement of network applications.

SF-FC has two purposes. One is not to waste the resources for useless traffic. The other is preventing network resource of adaptive flows being occupied by non-adaptive flows. This was achieved with lightweight processing and SF-FC showed higher performance than other current queuing models.

6 Discussions and Future Works

SF-FC is designed from the network administrator standpoint. It is not sufficient to use the protocol field of the IP header to identify non-adaptive and adaptive flows. This is because the field is controlled by end hosts and can be modified intentionally.

Therefore, we need to incorporate a function that identifies flows at the network level by monitoring the characteristic (inter-arrival time, duration time, and throughput) of the flows[3, 7].

Credit-Based Flow Control[6] and Rate-Based Congestion Control[8] are similar to SF-FC in that they conduct hop-by-hop flow control. They are different from SF-FC in that they aim to avoid any packet loss caused by buffer overflow over ATM networks. In SF-FC and our previous approach NBQ[11], the upstream router instantly discards a wasted packet of non-adaptive flows according to the link state information from the downstream routers. SF-FC can also consider fairness between non-adaptive and adaptive flows when allocating bandwidth.

7 Summary

As multimedia services using video and audio stream over the Internet has become manifold, various protocols have begun to run over IP. The key problem in such an environment is how we allocate limited bandwidth to flow with application QoS requirements. Such problems need to be considered through the provision of fairness and the effective utilization of the entire network.

However, most research for fair queuing tend to focus on reducing processing overhead and achieving exact fairness for each flow. In this paper, we claimed that perfect fair queuing is not effective for end users in the current network environment.

We have proposed Soft Fair Flow Control, which achieves fair network resource allocation to non-adaptive and adaptive flows. SF-FC allocates bandwidth by considering the flow state from the downstream routers. SF-FC avoids allocating network resource to wasted packets that will be dropped at the downstream router. As a result, SF-FC improves the performance of user application in a heterogeneous network environment.

Acknowledgment

The authors are grateful to members of the Keio RT-HDI Project for valuable comments and criticisms.

We thank Clifford Mercer and the anonymous NOSSDAV reviewers for several comments and suggestions that helped improve the quality of this paper.

This work is supported by Japan Society for the Promotion of Science Research Fellowships for Young Scientists.

References

[1] J. Postel (ed.). User Datagram Protocol. RFC 768, August 1980.

[2] J. Postel (ed.). Transmission Control Protocol – DARPA internet program protocol specification. RFC 793, September 1981.

[3] J. Padhye *et. al.* Modeling TCP throughput: A simple model and its empirical validation. In *Proceedings of ACM SIGCOMM'98*, September 1998.

[4] S. Blake *et. al.* An architecture for differentiated services. RFC 2475, December 1998.

[5] S. Floyd and K. Fall. Promoting the use of end-to-end congestion control in the internet. *IEEE/ACM Transactions on Networking*, 7(4), August 1999.

[6] H. T. Kung, T. Blackwell, and A. Chapman. Credit-based flow control for atm networks: Credit update protocol, adaptive credit allocation, and statistical multiplexing. In *Proceedings of ACM SIGCOMM'94*, May 1994.

[7] A. Mena and J. Heidemann. An empirical study of real audio traffic. In *Proceedings of IEEE INFOCOM'00*, March 2000.

[8] P. P. Mishra and H. Kanakia. A hop by hop rate-based congestion control scheme. In *Proceedings of ACM SIGCOMM'92*, August 1992.

[9] UCB/LBNL/VINT Network Simulator ns (version 2). *http://www-mash.cs.berkeley.edu/ns/.*

[10] M. Shreedhar and G. Varghese. Efficient fair queueing using deficit round robin. In *Proceedings of ACM SIGCOMM'95*, October 1995.

[11] Y. Tamura, Y. Tobe, and H. Tokuda. NBQ: Negihbor-state based queuing for adaptive bandwidth sharing. In *Proceedings of IEEE ICNP'99*, November 1999.

[12] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. RSVP: A new resource reservation protocol. *IEEE Network*, September 1993.