International Journal of Navigation and Port Research
Vol.28, No.5 pp. 299~303, 2004 (ISSN-1598-5725)

A Study on Fair Bandwidth Allocation in Core-Stateless Networks: Improved Algorithm and Its Evaluation

Mun-Kyung Kim · Kyoung-Hyun Seo** · Dong-Cheol Yuk*** · Seung-Seob Park*****

*, **, *** *Graduate School, PKNU, Busan Korea*
**** *Div. of Electronic, Computer and Telecommunication Engineering, PKNU, Busan Korea*

Abstract : *In the Internet, to guarantee transmission rate and delay and to achieve fair bandwidth allocation, many per-flow scheduling algorithms, such as fair queueing, which have many desirable properties for congestion control, have been proposed and designed . However, algorithms based on per-flow need maintain rate state, buffer management and packet scheduling, so that it cost great deal to implement.*

Therefore, in this paper, to implement routers cost-effectively, we propose CS-FNE algorithm based on FNE in Core-Stateless network. We evaluate CS-FNE comparing with four additional algorithms i.e., CSFQ, FRED, RED and DRR, in several different configurations and traffic sources. Through simulation results, we show that CS-FNE algorithm can allocate fair bandwidth approximately and is simpler and easier to implement than other per-flow basis queueing mechanisms.

Key words : *Fairness, Scheduling, CSFQ, FRED, RED, DRR, CS-FNE*

1. Introduction

Recently, due to the advance of data communication techniques, the Internet traffic grows rapidly. Internet services have changed from simply non real-time services to various services such as multimedia voice, real-time image data, on-line game, P2P, remote education and electronic commerce, so that more improved fairness, enlargement and security than existing services may be demanded in network.

For high speed of Internet networks, qualities of real-time and non real-time services, IETF(Internet Engineering Task Force) defines the multi-protocol label switching(MPLS) techniques. However, it costs great deal that all routers convert to MPLS routers and the networking method must be changed. In addition, the relay node controls the bandwidth as same manner, so that the cost of processing TCP/IP data is also expensive. Therefore, many papers have been studied Core-Stateless networks which distinguish between core nodes and edge nodes. Typically, there is CSFQ(Core-Stateless Fair Queueing) algorithm about allocating bandwidth fairly in the Core-Stateless network(Ion Stoica, Scott Shenker, and Hui Zhang).

As the growth of voice, video and game data, the network facilities have necessary to control the resources. In end-to-end congestion control, stability and efficiency are demanded and congestion control algorithms need to allocate bandwidth fairly. Until now, the fair allocation has been achieved by using per-flow queueing mechanism such as FQ(Fair Queueing), WFQ(Weighted Fair Queueing) and per-flow dropping mechanism such as FRED(Flow Random Early Drop). These mechanisms are more complex to implement than traditional FIFO queueing and make data processing slowly. Moreover, all nodes perform packet classification and per-flow congestion control, so that these mechanisms aren't suitable for high-speed networks.

Recently, to resolve the complicity, many methods have proposed various frameworks and mechanisms - Stoica proposed CSFQ in Core-Stateless network and Coa proposed RFQ(Rainbow Fair Queueing)(Ion Stoica, Scott Shenker, and Hui Zhang; Z. Cao, Z Wang, E. Zegura). The difference between CSFQ and RFQ is that CSFQ(with per flow) processes per-flow packets and assesses label i based on flow number, on the other hand, RFQ(without per flow) controls packets without per flow and assesses color label i based on packet average rate. However, RFQ computes the incoming packet rate by exponential distribution, so that simplicity is needed.

FNE(Flow Number Estimation) queueing mechanism, proposed by Li and Leu(2002), achieves computing packet rate by using hash function sorting and it can be easily implemented.

In this paper, to reduce the complexity and allocate bandwidth fairly, we apply FNE mechanism to Core-

*, **, *** Corresponding Author : Mun-Kyung Kim, geangee@hotmail.com 051)620-6389
**** parkss@pknu.ac.kr 051)620-6389

Stateless network. We propose CS-FNE mechanism and through the simulation result, we evaluate the performance with alternate approaches.

In Section 1 and Section 2, FNE and CS-FNE will be described in detail. Then simulation model and parameters are given in Section 3. We also evaluate the different existing schemes and show the results in Section 3. Finally, the conclusion remarks are given in Section 4.

2. Related Works

WFQ consider the propagation delay and fair bandwidth allocation. However, it complicated to implement. Although variational WFQ is proposed to achieve accuracy and minimize complexity, these mechanisms require maintaining per-flow state and classification. Therefore, the simplifying complexity is needed in high-speed networks which many flows pass through.

For improvement of simple drop-tail scheme, RED (Random Early Detection) is proposed. RED computes Q_{avg} (average queue size) per incoming packet and compares with predefined parameters, MAX_{th}(maximum threshold) and MIN_{th}(minimum threshold). However it have defects that parameters affect queueing mechanism sensitively and it does not guarantee the fairness. Although FRED proposed to improve RED makes the fairness better by using per-flow control, more packets in comparison with RED are lost when the average queue length exceed MAX_{th}. In this point of view, we propose the application to Core-Stateless network.

2.1 FNE(Flow Number Estimation) mechanism

FNE mechanism detects the number of flows occupying buffer space substantially. In spite of continuous data stream network has on-off states so that we can know the FIFO queue threshold as computing the number of incoming flows correctly; therefore fair bandwidth allocation is achieved. Basically, for ideal packet dropping, the summation of average rate haven't to be over C, as given by

$$\sum r_i \leq C \tag{1}$$

Assuming that n flows pass through the router and the total arrival rates is larger than the output link capacity, the packets of some mishaving flows must be dropped in order to control the queue length. Without sacrifice of generality, the arrival rates of all incoming flows are sorted to be $r_1 \leq r_2 \leq , \ldots , \leq r_n$. The optimal solution of CL(Cutting Line) exists in one of the n+1 regions, that is to say, $0 < CL \leq r_1$, $r_1 \leq CL \leq r_2 , \ldots , r_{n-1} \leq CL \leq r_n$ or $r_n \leq CL \leq \infty$. Accordingly, in $n+1$ flow, maximum CL is solution to

$$\sum_{i=1}^{n} \min(r_i, CL) = \frac{n \cdot CL + \sum_{i=1}^{n} r_i - \sum_{i=1}^{n} |r_i - CL|}{2} \tag{2}$$

When rates are over CL, packets will be dropped.

2.2 Measurement of Flow Rate

Trace ability and accuracy should be considered when flow rate is measured. FNE adopts the time sliding window(TSW) to measure the packet intensity. TSW algorithm determines dropping priority by considering whether average rate exceed aim rate specified in service profile of bound routers and marks packets in or out state. TSW measures the transmission rate per incoming packet and in accordance with window size, reflects the previous transmission rate to present transmission rate differently; so that TSW tolerates the burst data by average of summation with the pervious and present rate.

For recording the flow information and the packet header labeling, we use m-slot hash table; it computes the slot location by hashing function. When S_D_i (hash key value) is a pair of source and destination addresses and it is mapping on slot m, the solution of slot location in hash table by hashing function is given by

$$h(S\_D_i) = S\_D_i \bmod m. \tag{3}$$

The per-flow information management by hash map is more efficiency than using per-flow queue. However it cause the problems of using memory when flows are in same slot. Rehashing and chaining gives a solution; FNE adopts chaining method. When n flows pass through the router and the hash table has m slots, the probability which two maps are in one slot at least is given by

$$P_{collision} = 1 - \frac{m!/(m-n)!}{m^n} \tag{4}$$

To prevent collision, as shown in Fig. 1, chaining control informations in same slot by adopting linked-list; that is to say, it manages per-flow information with slot pointer. Flow 3 and Flow 4 are hashed by linked-list and exist in the same slot; $Src-Dst$ hash key value, the measured flow rate and the per-flow information in time t is stored in linked-list.

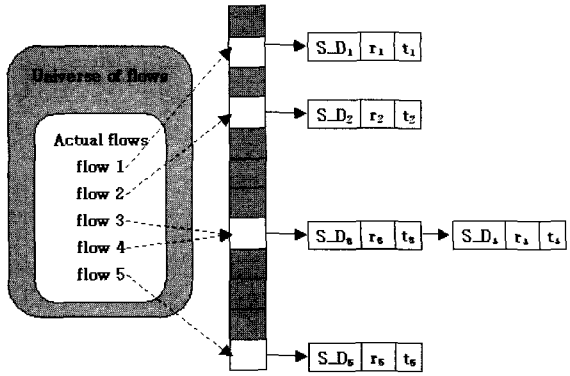

Fig. 1 The collision avoidance by the chaining method in hash table.

2.3 Estimation of Active Flow Numbers

The active flow number, N_{act} is estimated by comparing the incoming packet with the randomly selected packet in the queue. However, the packet based on this comparison method is not suitable for the practical network because the size of packets is not fixed. On the assumption that n flows pass through the router, r_i and p_i is estimated via the TSW and the buffer occupancy of flow i; p_i is the ratio of packets belonging the flow i in the queue. To compute N_{act}, we use two auxiliary rates r_{hit} and r_{miss}. Intuitively, r_{hit} is the flow rate that the flow ID of incoming packet matches that of one randomly selected packet in the queue and r_{miss} denotes the rate that two predescribed packets are not of the same flow.
They are shown as

$$r_{hit} = \sum_{i=1}^{n} r_i \cdot P_i \tag{5}$$

$$r_{miss} = \sum_{i=1}^{n} r_i \cdot (1 - P_i) \tag{6}$$

Then, the active flow number is defined as

$$N_{act} = \frac{r_{hit} + r_{miss}}{r_{hit}} \tag{7}$$

p_i is intuitively proportional to r_i in a FIFO queue. From the formulas Eq. (5) and Eq. (6), we can get

$$N_{act} = \frac{n}{C_r^2 + 1} \tag{8}$$

where C_r^2 is the coefficient of variation of the flow rate and equal to $\delta_r / E[r]$. In Fig. 2, *fair_rate* is solution to $link\_capacity / N_{act}$ as the threshold rate and it is updated per packet incoming.

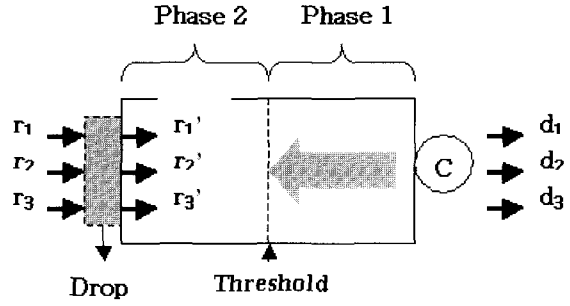

Fig. 2 FNE buffer management.

For example, if link has the capacity of 10 Mbps and the arrival rates, r_1, r_2 and r_3 are 2, 6 and 8, respectively, and fair_rate is 4, each flows receive bandwidth as min(r_1,4), min(r_2,4) and min(r_3,4), respectively. The probability of packet drop is given by

$$\begin{aligned} p_1 &= 1 - (4 / r_1) = 0.5 \\ p_2 &= 1 - (4 / r_2) = 0.33 \\ p_3 &= 1 - (4 / r_3) = 0 \end{aligned} \tag{9}$$

By these above processes, to control egress nodes and ingress nodes in Core-Stateless network, we implement FNE in Core-Stateless network and call CS-FNE.

2.4 Packet Labeling

Since the packet arrival rate is estimated, the packet label is assigned to the slot number of hash map; the packet label is between 0 and m when the packet has a probability of 1/m in m slot.

3. Simulation Environment and Performance Evaluation

All simulations are performed in NS-2(Network Simulation). To achieve the propriety of simulation, we consider the single congested link and multiple congested link in the same method as CSFQ and these are identical with reference(Ion Stoica, Scott Shenker, and Hui Zhang). We compare CSFQ's performance to four additional algorithms i.e., DRR(Deficit Round Robin), CSFQ, RED and FRED. DRR modified WFQ is per flow queueing mechanism which adopts round-robin and practices as benchmark even if it is ideal to allocate fair bandwidth.

We use the following parameters for the simulation. Each output link has the buffer of 64 KB, the propagation delay of 1 ms and the packet size of 1 KB. In the CSFQ, links have the buffer threshold of 16 KB and a constant K and

K_a are set to 100 ms equally; in the RED and FRED, links have the minimum threshold of 16 KB while they have maximum one of 32 KB. Win_len of TSW is set to 0.3 second.

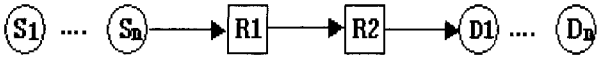

Fig. 3 The single congested link has the 10 Mbps capacity and the propagation delay of 1 ms.

In Fig. 3, we assume that the single congested link has 32 flows. In the first experiment, flows are all UDP's.

Average rates by the flow number are shown in Fig.4. DRR's performance is most ideal. Though the average of fair bandwidth increases as the flow number afterward, CS-FNE achieves more resonable degree of fairness; there is a little difference from CSFQ.

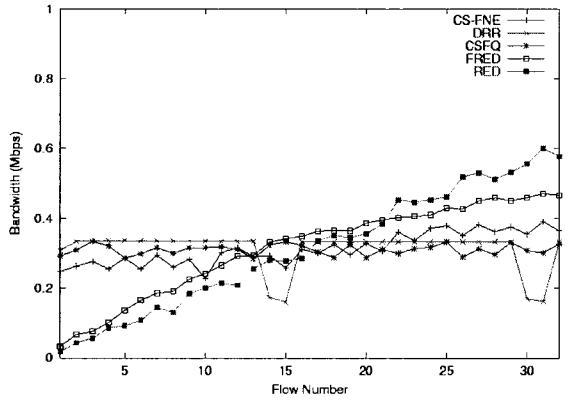

Fig. 4 The fair bandwidth of single congested link is shared by 32 UDP flows and has packet size of 1 KB.

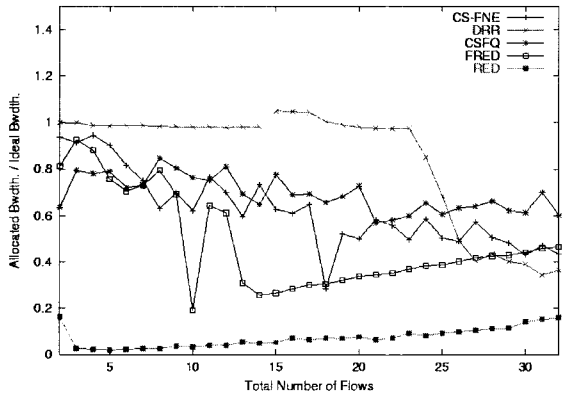

Fig. 5 The fair bandwidth of single congested link is shared by one TCP and 31 UDP flows and has packet size of 1 KB.

Simulation result is shown in Fig.5 on the assumption with link shared by one TCP and 31 UDP flows. When the TCP packet arrives the destination, receiver sends Ack message to sender responsively but UDP don't send Ack message and TCP has congestion mechanism such as slow-start and rapid recovery. Therefore, while bursty UDP packet occurs congestion in bottleneck, TCP is also affected; that is to say, TCP congestion recovers mechanism affects UDP flows. We simulate about it. CS-FNE vibrate a little than CSFQ but while flow number increases gradually, performance is more superior than RED, FRED and DRR. DRR's performance deteriorates from 22 flow number, though performance is more effective, because it is affected while TCP and UDP flows share the buffer together, differently from all UDP flows.

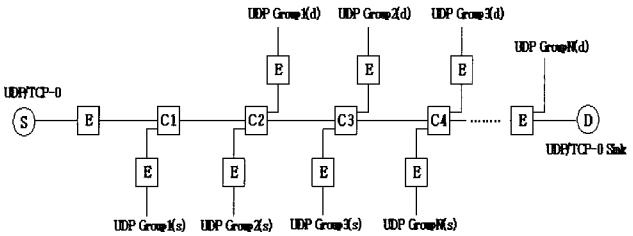

Fig. 6 The multiple congested link have capacity of 10 Mbps and propagation delay of 1ms.

In Fig. 6, GFC(Generic Fairness Configuration) network is constituted in the same way with Stoica's multiple congested link; each link has 10 Mbps capacity and 1 ms propagation delay. Only S source is replaced by UDP or TCP flow and the source UDP GroupN(s) is linked with the destination UDP GroupN(d) of the next node.

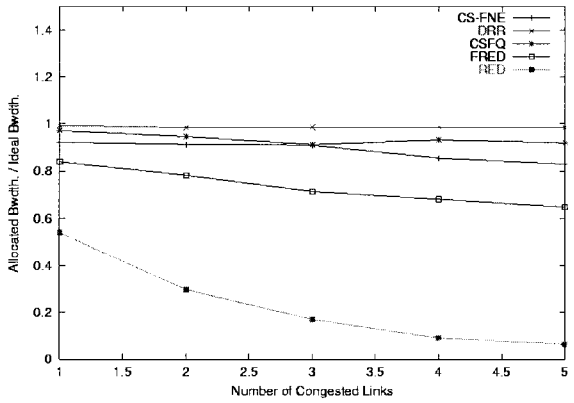

Fig. 7 The fair bandwidth of multiple congested link in which each link has ten cross flows(all UDP's), S=UDP and packet size=1 KB.

On the assumption that all ingress edge nodes, except initial node in the left, is formed 10 UDP sources per group,

Fig. 7 shows the simulation result based on the topology shown in Fig. 6. As shown in Fig. 7, DRR's performance is ideal and CS-FNE's performance shows minute difference from CSFQ. FRED is more efficient than the case of single link shared by all UDP flows. However, in RED, performance decline decease with increasing the number of congested link.

We use the topology shown in Fig. 6 for simulation of Fig. 8 and source of initial edge node on the left is replaced by TCP and per group has 10 UDP sources. Simulation result shows that CS-FNE's performance is more efficient on the fair bandwidth allocation than CSFQ between link number 2 and 3.5 but decline at 5. FRED's performance, comparing with the result shown in Fig. 7, has wide difference.

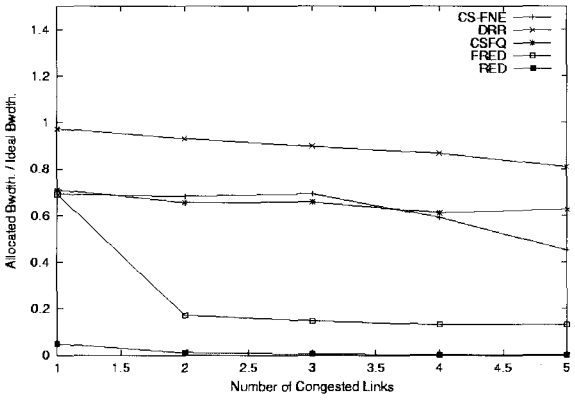

Fig. 8 The fair bandwidth of multiple congested link in which each link has ten cross flows(all UDP's), S=TCP and packet size=1 KB.

4. Conclusion

To guarantee transmission rate and propagation delay, many per-flow scheduling algorithms such as FQ, WFQ and FRED are proposed. However, algorithms based on per-flow scheduling are complex to implement and data processing rate is slow. Because rate state, buffer management and packet scheduling are required. Moreover, these algorithms perform packet classification and congestion control per flow in all nodes, so that they are not suitable for high speed network.

In this paper, to reduce complexity, we apply FNE queueing mechanism to Core-Stateless network. We call the mechanism CS-FNE and evaluate it comparing with CSFQ, FRED, RED and DRR. In the single and multiple congested link, CS-FNE's performance is more effective than RED and FRED. However, though fruit of ideal fairness is not proposed, it doesn't adopt the exponential distribution to compute the fair rate, so it is meaningful to implement easily and cost-effectively by hardware.

From now, to get more effective solution, improved fairness and congestion control with TCP data are necessary.

References

[1] Demers, A., Keshav, S., and Shenker, S., (1990) "Analysis and simulation of a fair queueing algorithm," J. Internetw. Res. Experience, pp. 3-26.

[2] Yuk, Dong-Cheol, Park, Seung-Seob, (2003) "A Study on Fair Bandwidth Allocation Based on FNE Algorithm in Core-Stateless Network," Korean Institute of Navigation and Port Reaserch, Fall Conference, pp.77-82 (Korean).

[3] Clark, D. D. and Fang, W.(1998), "Explicit allocation of best-effort packet delivery service," IEEE Trans, 362-373.

[4] Lin, D. and Morris, R., (1997) "Dynamics of random early detection," in Proc. ACM SIGCOMM, Cannes, France, pp. 1427-137.

[5] Ion Stoica, Scott Shenker, and Hui Zhang, "Core-Stateless Fair Queueing: Achieving Approximat -ely Fair Bandwidth Allocation in High Speed Networks," in Proceeding of SIGCOMM'98, Oct, 1997.

[6] Li, Jung-Shian and Leu, Ming-Shiann, (2002) "Fair bandwidth share using flow number estimation," Communications, 2002. ICC 2002. IEEE International Conference on, Vol. 2, pp. 1274-1278.

[7] Zhang, L.(1990), "Virtual clock: a new traffic control algorithm for packet switching networks," in Proc. ACM, SIGCOMM 90, pp. 19-29.

[8] Shreedhar, M. and Varghese, G., (1996) "Efficient fair queueing using deficit round robin," IEEE/ACM Trans. Networking, pp. 375-385.

[9] NS simulator, available from http://www.isi.edu/nsnam/ns.

[10] Parekh, A. A generalized processor sharing approach to flow control.

[11] Floyd, S. and Jacobson, V., (1993) "Random early detection for congestion avoidance," IEEE/ACM Trans. Networking, Vol. 1, pp. 397-413.

[12] Cao, Z., Wang, Z., Zegura, E., (2000) "Rainbow fair queueing: fair bandwidth sharing without per-flow state", Proceedings INFOCOM. pp. 922-931.

Received 30 April 2004
Accepted 17 June 2004