Preferential Bandwidth Allocation for Short Flows with Active Queue Management

Zhang Heying, Liu Lu, Xiao Liquan, Dou Wenhua

School of Computer, National University of Defense Technology, 410073,
Changsha, Hunan, China
hey_zhang@hotmail.com, douwh@vip.sina.com

Abstract. Several fair queueing mechanisms based on stateless core (SCORE)/dynamic packet state (DPS) architecture have been proposed to address the scalability problem of stateful architectures. However, most of these mechanisms indiscriminatingly label every packet in edge routers while only a small fraction of the packets that come from fast flows will be dropped by core routers. Moreover, these mechanisms usually apply simple techniques to detect congestion, which makes them unable to control the queue length. In this paper, a new fair bandwidth allocation mechanism is proposed. In the new mechanism, edge routers only label the packets of long flows so that the bandwidth is preferentially allocated to short flows and the remaining is fairly allocated among the competing long flows. Furthermore, routers can keep the queue length at a reference value using active queue management (AQM) algorithm. The simulation results show that this mechanism performs well in many aspects.

1 Introduction

In current Internet, routers simply forward each incoming packet to its destination, regardless of which flow it belongs to. Therefore, some greedy, unresponsive flows will unfairly obtain more bandwidth than the conservative, responsive ones when congestion occurs. Such a situation will probably cause the danger of congestion collapse and the starvation of conformant flows.

The stateless core architecture, or SCORE for short, is proposed to achieve approximate fairness and reasonable scalability simultaneously. The key technique used to implement the SCORE network is the dynamic packet state (DPS), which inserts

the flow state information into the header of packets. In the SCORE/DPS network architecture, routers are divided into edge routers and core routers. Edge routers maintain per-flow state and insert it into the header of the incoming packet [1-4]. Core routers use the simple first-in first-out (FIFO) queueing and drop the incoming packet based on the state information carried in its header when congestion occurs.

Unfortunately, the existing mechanisms based on SCORE/DPS architecture have the following limitations. First, they label every packet passing through edge routers, which is not really necessary since only packets of high-bandwidth flows will be dropped probabilistically when the network becomes congested. Moreover, it is showed by recent measurement that most of the traffic is actually carried by a small number of flows, while the large remaining amount of flows is very small both in size and lifetime [5,6]. So it is reasonable to just maintain the state of these minority flows that tend to occupy more bandwidth than others and label their packets. Second, these mechanisms treat short flows and long flows equally. In fact, the throughput and delay of the short-lived TCP (Transmission Control Protocol) flows will deteriorate severely when competing with the long-lived flows due to lack of sufficient packets to activate duplicate acknowledgments and the dependence on timeout to detect packet loss. Although several approaches have been proposed to deal with short flows preferentially, they cannot allocate bandwidth fairly among the competing long flows [7,8]. Third, to the best of our knowledge, none of these proposed SCORE/DPS mechanisms applies specific approach to control the queue length, which corresponds to the queueing delay experienced by the backlogged packets.

In order to address these issues, we propose a new fair bandwidth sharing mechanism in this paper. The features of the new mechanism include simplifying the operation of routers, protecting short flows and achieving fairness among long flows. We use a well-designed AQM (Active Queue Management) algorithm, called proportional integral based series compensation and position feedback compensation (PIP), to detect congestion and control queue length [9]. So the proposed mechanism is called FPIP (fair PIP).

The rest of the paper is organized as follows. In section 2, we describe FPIP in detail, including the core router and edge router. In section 3, we evaluate the performance of FPIP through extensive simulations. Finally, we conclude in section 4.

2 FPIP Framework

In this section, we present FPIP, a packet labeling and queue management mechanism that significantly simplifies the operation of routers without affecting the performance by taking into account the ubiquitous heavy-tailed distribution of the Internet traffic. We apply the network model comprised of edge routers and core routers. For each active flow, the edge routers maintain a traffic counter that tracks how many bits have been observed so far and determine whether the flow is short or long. When a packet comes from a short flow, the traffic counter of the flow increases. Otherwise, the flow rate is estimated and inserted into the header of the packet. The routers estimate the aggregate arrival rate of short flows and the number of active long flows, and then calculate the fair share based on them. In addition, a notable feature of FPIP is the use of AQM algorithm in detecting congestion and controlling the queue length, from which the delay-sensitive applications such as Web or Telnet can benefit.

2.1 Estimating the Flow Arrival Rate

To protect short flows, we should distinguish them from long flows at first and then decrease their loss rates. In our mechanism, the edge router maintains a traffic counter for each active flow, which is used to record the number of the bits sent by this flow. Once the traffic counter exceeds a certain "bit threshold", noted as *bitThresh*, the flow will be considered long. Otherwise, it is considered short. For the long flows, the edge routers estimate their arrival rates and label their packets. Instead, for the short flows, only their traffic counters increase.

We use the exponential averaging formula to estimate the long flow arrival rate. Let Δt_i^k be the time interval between the k^{th} and the $(k\text{-}1)^{th}$ packet of flow i. The estimated rate of flow i is calculated as

$$r_i^{new} = (1 - e^{-\Delta t_i^k / K_r}) l_i^k / \Delta t_i^k + e^{-\Delta t_i^k / K_r} r_i^{old} \quad . \qquad (1)$$

where l_i^k is the length of the k^{th} arrival packet of flow i and K_r is a constant.

2.2 Estimating the Aggregate Arrival Rate of Short Flows

To calculate the bandwidth that can be allocated to long flows, we should estimate the aggregate arrival rate of short flows at first. For each arrival packet, the router checks

its label to see which kind of flow it comes from. If the packet label equals to zero, the packet is thought of as coming from short flow. Let l be the length of the arrival packet and Δt be the inter-arrival time of the consecutive packets that come from short flows. The router calculates the aggregate arrival rate of short flow, denoted by *sRate*, as follows

$$sRate = (1 - e^{-\Delta t^k / K_s}) l^k / \Delta t^k + e^{-\Delta t^k / K_s} sRate \quad . \tag{2}$$

where K_s is a constant.

If the label of the arrival packet is greater than zero, the packet is thought of as coming from long flow. *sRate* is also updated according to (2), where l equals to zero. By doing so, *sRate* will reflect the real aggregate arrival rate of short flows even if there have been no packets from short flows for a long period of time.

Now, the bandwidth that can be obtained by long flows is readily available:

$$C_l = \max\{0, C - sRate\} \quad . \tag{3}$$

2.3 Estimating the Number of Active Long Flows

In FPIP, the routers calculate the fair share rate based on two variables: the bandwidth allocated to long flows and the number of the active long flows (*Nactivel*). The former has been determined easily, while the latter is a lot harder to estimate. Several approaches have been proposed previously to address this issue [10-12]. Since these approaches are motivated by some specific goals, none of them can be copied here.

In this paper, we introduce a new method to estimate the number of the active long flows. According to our method, the router is required to maintain a state table for tracking the arrival time (denoted by *prevtime*) of the packet that has lately arrived from each long flow. For each arrival packet, if its label is greater than zero, the *prevtime* of the corresponding flow in the state table is checked. If it equals to zero, the number of active long flows increases and the *prevtime* is set to the current time. Otherwise, only the *prevtime* is replaced by the current time. In order to estimate the number of the flows sharing the bandwidth during a longer period of time rather than that of the flows currently having packets in the buffer, the flow table is not updated when there is packet leaving the queue. Instead, it is updated periodically with a constant frequency, which can be viewed as a background task, for it is shifted from the high-speed data-forwarding path. When the update timer expires, entries of the table

are checked one by one. If the interval between the current time and the *prevtime* of a flow is greater than a certain threshold (*Tn*), which means there is no packet from that flow in the last *Tn* time units, the flow is considered terminated. Thus, *Nactivel* is reduced and the *prevtime* of the flow is reset to zero.

2.4 Estimating and Adjusting the Fair Share Rate

The problem of the fair bandwidth sharing occurs along with the presence of the network congestion, and the estimation of the fair share rate depends further on it. Therefore, it is of great importance to correctly detect congestion. In this paper, we apply AQM algorithm in congestion detection for the following reasons: (1) The packet drop probability calculated by AQM is a good representation of the congestion degree; (2) AQM algorithm is able to detect congestion and control queue length simultaneously. In our mechanism, we use a robust AQM algorithm called PIP [9].

The packet drop probability $p(k)$ determined by PIP is regarded as a measure of congestion. When $p(k)$ is greater than a random variable, it is likely that the link is congested and the fair share rate (R_{fair}) should be calculated. Let the capacity of the output link be C. Suppose *sRate* is less than C. R_{fair} is calculated as

$$R_{fair} = \frac{C - sRate}{Nactivel} \qquad Nactivel > 0 \quad . \tag{4}$$

When $p(k)$ is less than a random variable, the link is considered uncongested. To avoid under-utilization of the link, when the estimated rate of the accepted traffic (*cRate*) is less than the output link capacity C, the fair share rate is adjusted as follows

$$R_{fair}^{new} = \min\{C, \frac{C}{cRate} R_{fair}^{old}\} \quad . \tag{5}$$

The accepted rate is also estimated by exponential averaging:

$$cRate = (1 - e^{-\Delta t^k / K_c}) l^k / \Delta t + e^{-\Delta t^k / K_c} cRate \quad . \tag{6}$$

The implications of the parameters in (6) are similar to those in (1). Now, the incoming packet of the long flow will be dropped with the following probability

$$p\text{rob} = \max\{0, \frac{r_i - R_{fair}}{r_i}\} \quad . \tag{7}$$

3 Simulations

In this section, we use NS simulator to evaluate the performance of FPIP and compare with CSFQ and RED [13]. In the simulations, we use the network topology with multiple congested links shown in Fig.1. The number of congested links varies from one to five. The capacities of all the access links are 30 Mbps, and those of the congested links are 10 Mbps. The propagation delay of each link is 5 ms. Each router is connected with five UDP flows which terminate at the next router and send at 4 Mbps. Thus, all the links between neighboring routers are congested.

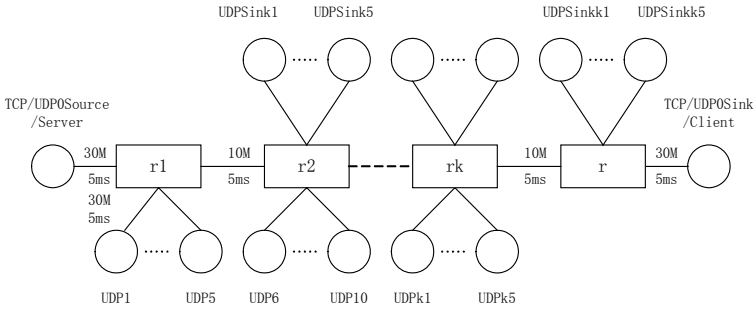

Fig. 1. Network topology used in the simulations

In the first experiment, a TCP flow traverses all the congested links. Fig.2 shows the bandwidths achieved by the TCP flow as a function of the number of congested links. We compare the bandwidth achieved by each flow through the normalized bandwidth, which is defined as the ratio of the allocated bandwidth to the ideal bandwidth. In RED, the TCP flow is submerged by the high-speed unresponsive flows. The TCP flow achieves more bandwidth under FPIP than that under CSFQ. Fig.3 shows the queue dynamics in router r under CSFQ and FPIP when the number of the congested links is 5. For CSFQ, the queue length of each router is about 300 Kbytes in steady state, while for FPIP, the queue length is 25 Kbytes.

In the second experiment, the TCP flow is replaced by a UDP flow (denoted by UDP0 in Fig. 1) sending at its fair share rate 1.67 Mbps. Fig. 4 shows the normalized bandwidth achieved by UDP0. Similarly, RED has the worst performance. FPIP performs slightly better than CSFQ.

In the last experiment, a Web flow traverses all the congested links. Fig. 5 shows the response time of the Web flow traversing different numbers of congested links. We cannot show the result under RED, for the client of the Web flow cannot even re-

ceive a single response from the server. For the other two mechanisms, when the number of the congested links increases, the response time of the Web flow also increases. Moreover, the increase under CSFQ is faster than that under FPIP.

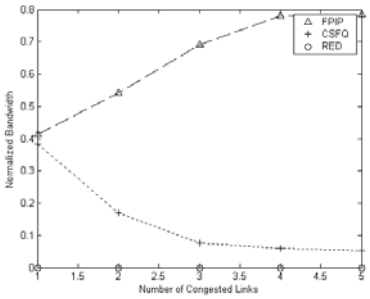

Fig. 2. Bandwidth achieved by TCP flow

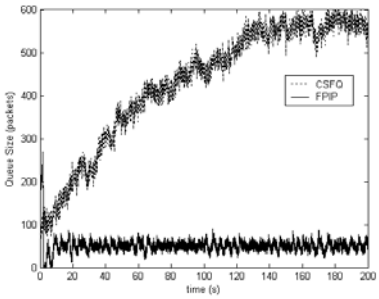

Fig. 3. The queue dynamics of CSFQ and FPIP

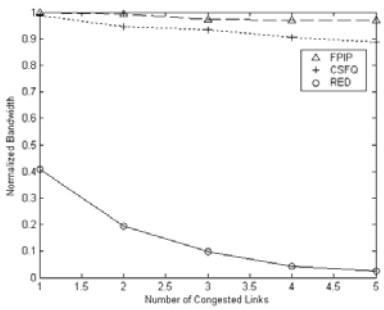

Fig. 4. Bandwidth achieved by UDP flow

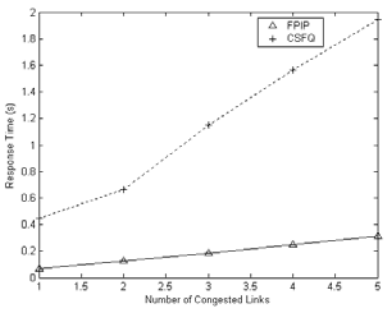

Fig. 5. The response time of web flow

4 Conclusions

In this paper, we present a new fair bandwidth allocation mechanism called FPIP. By labeling only the packets of long flows at edges, the mechanism greatly reduces the amount of flow state required and the processing done on it. In the core of the network, routers only drop packets from long flows with probability, while short flows will not be dropped as long as the capacity of the output link will satisfy their bandwidth demand. Furthermore, to provide low delay service, the core routers apply AQM mechanism to detect congestion and control queue length, which is beneficial to the adaptive flows and delay-sensitive applications. The results of simulations

show that FPIP can obtain approximate fairness among long flows, keep queue length at a desired value and reduce the response time of Web flows.

References

1. Stoica, I., Shenker, S., Zhang, H.: Core-stateless Fair Queueing: Achieving Approximately Fair Bandwidth Allocations in High Speed Networks. In Proceedings of ACM SIGCOMM 1998, Vancouver (1998) 118–130
2. Cao, Z., Wang, Z., Zegura, E.: Rainbow Fair Queueing: Fair Bandwidth Sharing Without Per-flow State. In Proceedings of IEEE INFOCOM 2000, Tel-Aviv, Israel (2000) 922-931
3. Clerget, A., Dabbous, W.: Tag-based Fair Bandwidth Sharing for Responsive and Unresponsive Flows. In Proceedings of IEEE INFOCOM 2001, An-chorage, AK (2001)
4. Ngin, H.T., Tham, C.K.: A Control-Theoretical Approach for Achieving Fair Bandwidth Allocations in Core-Stateless Networks. Computer Networks, Vol.40, (2002) 727-741
5. Mahajan, R., Floyd, S.: Controlling High Bandwidth Flows at the Congested Router. AT&T Center for Internet Research at ICSI (ACIRI), TR-01-001 (2001)
6. Brownlee, N., Claffy, K.C.: Understanding Internet Traffic Streams: Dragonflies and Tortoises Brownlee. IEEE Communications Magazine, Vol.40, (2002) 110-117
7. Zhang, Y., Qiu, L., Keshav, S.: Speeding up Short Data Transfers: Theory, Architecture Support, and Simulation Results. In Proceedings of NOSSDAV 2000, Chapel Hill, NC, USA (2000)
8. Guo, L., Matta, I.: The War Between Mice and Elephants. Technical Report BU-CS-2001-005, Boston University (2001)
9. Zhang, H.Y., Liu, B.H., Dou, W.H.: Design of a Robust Active Queue Management Algorithm Based on Feedback Compensation. In Proceedings of ACM SIGCOMM 2003, Karlsruhe, Germany (2003) 277-286
10. Lin, D., Morris, R.: Dynamics of Random Early Detection. In Proceedings of ACM SIGCOMM 1997, Cannes, France (1997) 127–137
11. Li, J.S., Leu, M.S.: Network Fair Bandwidth Share Using Hash Rate Estimation. Networks, Vol. 40, (2002) 125-141
12. Ott, T.J., Lakshman, T.V., Wong, L.: SRED: Stabilized RED. In Proceedings of IEEE INFOCOM 1999, New York, USA (1999) 1346-1355
13. Floyd, S., Jacobson, V.: Random Early Detection Gateways for Congestion Avoidance. IEEE/ACM Transactions on Networking, Vol.4, (1993) 397-413