Journal of Zhejiang University SCIENCE
ISSN 1009-3095
http://www.zju.edu.cn/jzus
E-mail: jzus@zju.edu.cn

An adaptive mechanism to guarantee the bandwidth fairness of TCP flows

ZHANG Shun-liang (张顺亮), YE Cheng-qing (叶澄清)†

(*College of Computer Science & Engineering*, *Zhejiang University*, *Hangzhou 310027*, *China*)

†E-mail: ycq@mail.hz.zj.cn

Received June 4, 2003; revision accepted Sept. 2, 2003

Abstract: End-to-end TCP (transmission control protocol) congestion control can cause unfairness among multiple TCP connections with different RTT (Round Trip Time). The throughput of TCP connection is inversely proportional to its RTT. To resolve this problem, researchers have proposed many methods. The existing proposals for RTT-aware conditioner work well when congestion level is low. However, they over-protect long RTT flows and starve short RTT flows when congestion level is high. Due to this reason, an improved method based on adaptive thought is proposed. According to the congestion level of networks, the mechanism can adaptively adjust the degree of the protection to long RTT flows. Extensive simulation experiments showed that the proposed mechanism can guarantee the bandwidth fairness of TCP flows effectively and outperforms the existing methods.

Key words: Bandwidth allocation, Fairness guarantee, Congestion control, RTT, Adaptive method
doi:10.1631/jzus.2004.1361 **Document code:** A **CLC number:** TP393

INTRODUCTION

In current Internet, most of traffic is generated by traditional data transfer applications such as HTTP (hyper text transmission protocol), FTP (file transmission protocol), or SMTP (small mail transfer protocol). Usually, these elastic applications are not sensitive to the delivery time of individual packet, but rather to the total transfer time of the data. They often use TCP, which provides reliable end-to-end control over the "best-effort" service of IP. Because the throughput of a connection greatly depends on its bandwidth utilization of bottleneck, it is necessary for connections to achieve fair resource sharing at bottleneck links. Unfortunately, with TCP Reno widely used in the Internet, the throughput of a connection greatly depends on its RTT, which prevents fair sharing of bandwidth at the bottleneck link (Jeonghoon *et al.*, 1999). To solve this problem, the ideal congestion control mechanism is required to allocate the bandwidth of bottleneck link according to the number of connections sharing the link. Because TCP is an end-to-end control protocol, TCP end hosts have no means to get precise network information such as the number of connections, the bandwidth of the bottleneck, the RTT of connections. TCP cannot control the transmission rate according to the information. Therefore, a network support for congestion control is indispensable.

FAIRNESS ISSUE OF TCP AND THE RELATED WORK

TCP has the following two properties, which

cause the bandwidth fairness problem (Golestani and Sabnani, 1999): (1) TCP is the traffic regulation algorithm to keep the total outstanding packets below the allocated window size; (2) A TCP sender updates its congestion window size every RTT. These properties cause a non-uniform increase of the rate with different RTT, because a connection with shorter RTT updates its window value faster than a connection with longer RTT. In other words, the connection with smaller RTT suppresses the connection with the longer RTT. Furthermore, Seddigh *et al.*(1999) found that many factors affect the bandwidth acquired by a TCP flow in both over-provisioned and under-provisioned networks. These factors include RTT, the number of micro-flows in a target aggregate, the size of the target rate, packet size and existence of non-responsive flows. Some factors can be explained via the following equation of Mathis *et al.*(1997).

$$BW \propto \frac{MSS}{RTT\sqrt{p}} \tag{1}$$

To solve the bandwidth fairness problem of TCP, researchers proposed several TCP rate control schemes for the fair-share of bottleneck bandwidth. In this paper, we only investigate the fairness problem caused by RTT. Nandy *et al.*(2000) first proposed an RTT-aware traffic conditioner to mitigate this unfairness. The traffic conditioner avoids RTT bias of TCP connections through marking packets with high drop priority inversely proportional to the square of the RTT according to the steady state TCP behavior. Matsuda *et al.*(2002) proposed a method using ECN-capable TCP, which adaptively marks the packets according to the queue size and RTT of each connection passing through the router. The aforementioned methods work well when the congestion level is low, but they over protect the long RTT flows and starve the short RTT flows when the congestion level is high. Combining BLUE (Feng *et al.*, 2002), we implement a scheme with the same theory as the aforementioned methods in ns-2 simulator to investigate their performance. The network topology is shown in Fig.1. The experimental results are shown in Fig.2 and Fig.3 (The RTT of connections is proportional to their ID number). Habib *et al.*(2002) also encountered the same problem and proposed a scheme to solve it. However, our experiments showed that this scheme could not solve this problem perfectly.

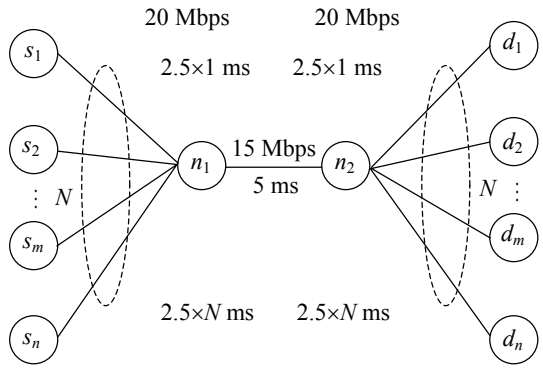

Fig.1 Network topology

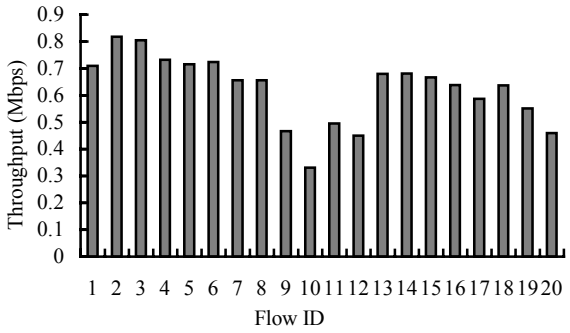

Fig.2 Bandwidth acquired by flows under low level congestion condition

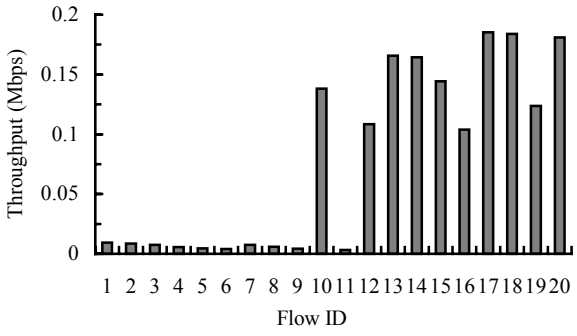

Fig.3 Bandwidth acquired by flows under high level congestion condition

MECHEMISM TO GUARENTEE THE BANDWIDTH FAIRNESS

The aforementioned methods work well when the congestion level is low, because Eq.(1) accurately represents the fast retransmission and recovery behavior of flows. However, when the congestion level is high, short RTT flows time out because the conditioner only protects long RTT flows after satisfying the target rate. Besides, it is more difficult to estimate RTT of TCP flows accurately when the congestion is severe because of queuing delay, packet loss, and the oscillation of delay, etc. cause RTT to fluctuate frequently. Furthermore, when congestion level is low, only few packets are dropped due to congestion. However, when congestion is severer, large quantity of packets are dropped. Therefore the short RTT flows send more packets and lose more packets also. So they lose some superiority over long RTT flows during bandwidth competition process.

To remedy this situation, we can take some strategies. The first one is to adopt a more accurate model (Padhye *et al.*, 1998), which considers round trip time-out (RTO). Eq.(2) shows this approximation.

$$BW \approx \frac{1}{RTT\sqrt{\frac{2bp}{3}} + To \times \min(1, 3\sqrt{\frac{3bp}{8}})\, p(1+32p^2)} \tag{2}$$

The method proposed by Habib is based on this model and can improve the performance of the aforementioned methods due to considering of RTO. However, there are still some factors limiting its application. Parameter b is not equal in different TCP variants. RTO is estimated based on RTT, so the error during estimating of RTT is enlarged when calculating RTO. Besides, transmission RTO information from end hosts to routers will increase the overhead furthermore. Therefore, we consider the second strategy, which can adaptively adjust the degree of protection to long RTT flows according to the congestion level of the network. As mentioned before, the existing methods work well when congestion level is low but their performances degrade greatly when congestion is severe. So we consider providing long RTT flows with different degree of protection under various congestion levels. In other words, when congestion level is low, we give them strong protection. With the increase of the congestion, the protection to long RTT flows decreases gradually. Congestion level can be measured based on queue length or the arrival rate of packets at routers. Though different active queue management (AQM) methods measure congestion in different ways, most AQM mechanisms will increase the marking probability when congestion level increase. So, we can infer the congestion level from p (marking probability) of the AQM mechanism at routers. Using this idea we design an improved method.

In our case, the sending host using TCP-Reno gives the edge hint of the lowest RTT recorded. The RTT is computed using the base RTT computation from TCP Vegas code. We define a 16-bit integer in the reserved space of IP header to transmit RTT information. But in practice, 9 bits is enough for coverage of up to 80% of RTT typically observed (Allman, 2000). To improve the accuracy of the estimated RTT, the average RTT is calculated using the exponential weighted moving average method. It is expressed by the following equation.

$$RTT_{\text{avg}}^{n} = (1 - w_q) RTT_{\text{avg}}^{n-1} + w_q RTT \tag{3}$$

where $RTT_{\text{avg}}^{n}, RTT_{\text{avg}}^{n-1}$ are average RTT of current and the last time respectively; w_q is the weight, which was set as 0.1 in the experiment. This is a trade-off between the sensitivity and stability. We implemented this idea based on an active queue management mechanism–BLUE, which uses packet loss and link idle events to manage congestion. There is little literature on its fairness problem due to the RTT difference of TCP flows, so we investigate BLUE. The main idea of BLUE is shown as the following pseudo code:

Upon packet loss (or $q_{\text{len}} > q_{\text{lim}}$) event:

If ((*now*–*last _update*)>*freeze_time*) then

$p_{\text{mark}}=p_{\text{mark}}+det_{\text{in}}$

last _update=now

Upon link idle event:

If ((*now*–*last_update*)>*freeze_time*) then

$p_{\text{mark}}=p_{\text{mark}}-det_{\text{de}}$

last_update=now

Because marking probability p is important to our method, it is estimated in the same way as computing RTT. Besides, we calculate the arithmetical average RTT of all the flows passing through the bottleneck link. The marking probability of BLUE is regulated with the following equation:

$$P_i=\left(\frac{RTT_{\text{savg}}}{RTT_i}\right)^{\text{alpa}}P_b \tag{4}$$

where P_i is the marking probability of flow i after regulation; RTT_i is the average RTT of flow i calculated with Eq.(3); RTT_{savg} is the arithmetical average RTT of all flows; P_b is the marking probability of BLUE; The parameter *alpa* is calculated with the following equation.

$$alpa=\begin{cases}2.0 & \text{if } P_b^{\text{avg}}\le 0.02\\ \dfrac{P_{\text{th}}}{P_b^{\text{avg}}} & \text{if } P_b^{\text{avg}}>0.02\end{cases} \tag{9}$$

where P_b^{avg} is the average marking probability of BLUE, which is calculated in the same way as RTT; P_{th} is the threshold value, which determines the degree of protection to long RTT flows. If P_{th} is too small, it can avoid over protecting long RTT flows when congestion level is high, but it cannot give long RTT flows enough protection when congestion level is low. However, if it is too big, it will over protect long RTT flows when congestion level is high. P_{th} was set as 0.05 in the experiments. In this way, the degree of protection to the long RTT flows decreases gradually with the increase of congestion level. In fact, the proposed method is also based on Eq.(1). Due to the reasons mentioned before, we just make some regulation.

PERFORMANCE EVELUATION

To prove the effectiveness of the method proposed here, we made extensive simulation experiments with ns-2. All the experiments in this paper implemented ECN (explicit congestion notification) mechanism. For convenience of comparison, we denote the method proposed by Matsuda as MRTT, original BLUE mechanism as BLUE, the proposed method as ZRTT, the method proposed by Habib as PRTO, the method proposed by Stoica as CSFQ (Stoica *et al.*, 1998). The experiment network topology is shown in Fig.1. There is a bottleneck link between node n_1 and n_2. The number of connections is N. The number of sessions in each connection is variable. With the increase of connection ID number, the delay of the connection also increases, so does the RTT of TCP flows passing through it.

The configuration parameter of BLUE follows the recommended settings: *freeze_time*=50 ms, det_{in}=0.02, det_{de}=0.002, q_{lim}=100 (packet). Under the same experiment conditions as before, the results of ZRTT and BLUE are shown in Fig.4 and Fig.5.

Fig.4 shows that BLUE cannot overcome the shortcoming of TCP. The bottleneck bandwidth that each flow can acquire greatly depends on its RTT. The larger its RTT is, the less bandwidth a flow can acquire. From Fig.5, we can find that ZRTT can solve the fairness problem effectively.

In order to test the performance of the proposed method systematically, we did two sets of experiments in addition. To compare the performance of aforementioned methods, we used Jain's Fairness Index (Chiu and Jain, 1989). The index quantifies the degree of similarity among all the flow bandwidths. Given the set of throughputs (x_1, x_2, …, x_n), the fairness index is calculated as follows:

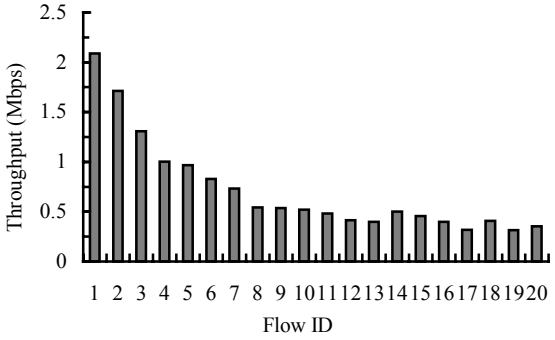

Fig.4 Bandwidth acquired by flows with BLUE

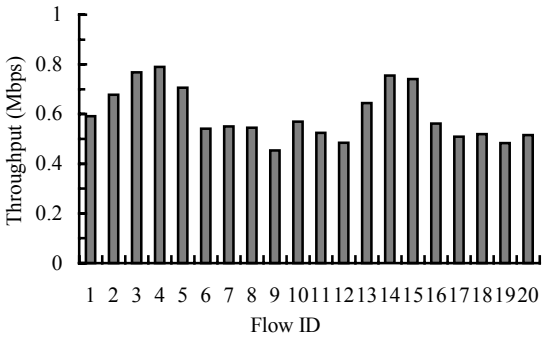

Fig.5 Bandwidth acquired by flows with ZRTT

$$f(x_1, x_2, \ldots x_n) = \frac{(\sum_{i=1}^{n} x_i)^2}{n\sum_{i=1}^{n} x_i^2} \quad (6)$$

The fairness index always lies between 0 and 1. Hence, a higher fairness index indicates better fairness between flows. The fairness index is 1 when all the throughputs are equal. Otherwise, the fairness index drops below 1.

In the first set of experiments, the performance of these methods under various RTT difference condition was investigated. In this set of experiments, there was only one session in each connection. With the increase of the number of TCP connections, the difference of the RTT of TCP flows increases also. The experiment's result is shown in Fig.6.

From Fig.6, we can find that the fairness provided by the three improved methods was good and only decreased a little with the increase of RTT difference. In this situation all three improved methods worked quite well and provided the flows with almost ideal fairness. Comparatively, the performance of PRTO is a little worse than that of the other two methods. Without any strategy to regulate the traffic, the fairness provided by BLUE decreases rapidly with the increase of RTT difference. Comparatively, CSFQ can greatly improve the bandwidth fairness among TCP flows. However, it cannot completely solve the bandwidth fairness problem of TCP flows due to RTT difference.

In the second set of experiments, the performance of these methods under different congestion level was investigated. In the first set of experiments, due to the large RTT difference of these flows and the fact that there was only one session in each connection, the congestion level was rather limited. Therefore, in the second set of experiments, we set the number of TCP connections to 20 and let the number of sessions in each connection increase gradually. So the congestion level at the bottleneck also increased. The experiment result is shown in Fig.7.

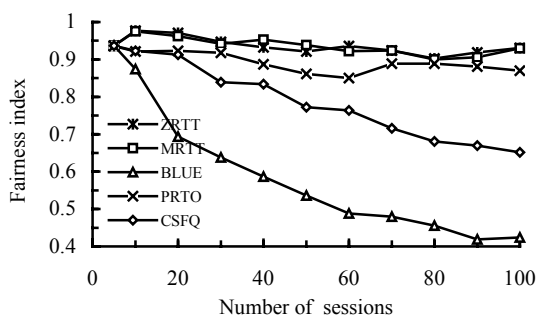

Fig.6 Fairness index vs the number of sessions

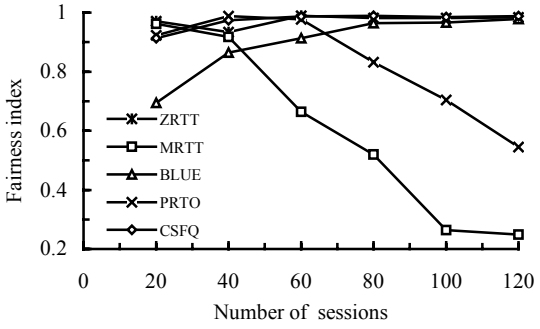

Fig.7 Fairness index vs the number of sessions

It was not difficult to find that the fairness provided MRTT decreased rapidly with the increase of the congestion level. The performance of PRTO was not yet perfect though it outperformed MRTT. The reason was that they over protect the long RTT flows. Compared with MRTT and PRTO, ZRTT worked quite well with the increase of the congestion level. The performance of ZRTT was as good as that of CSFQ, used exponential averaging to estimate the rate of a flow both at edge routers and core routers, which need considerable computation. But ZRTT only needed simple computation at routers. It was also a scalable method like CSFQ. CSFQ can effectively provide approximate fairness under certain conditions with unresponsive flows (UDP). However, ZRTT only aimed at the fairness problem among TCP flows due to RTT difference. Besides, we can find an interesting phenomenon from Fig.7. With the increase of the congestion level, the fairness index of BLUE increases gradually. In other words, the fairness problem of TCP degrades gradually with the increase of the congestion level. Because of the reason that we have mentioned, the short RTT flows lose their priority over long RTT flows gradually during the process of competition for bottleneck bandwidth, so the difference among the bottleneck bandwidths acquired by TCP flows decreases.

CONCLUSION

In this paper, we analyze the bandwidth fairness problem of TCP and point out the defect of existing methods. Based on the analysis and the related network model, an improved method based on adaptive thought is proposed. Extensive simulation experiments showed that the proposed method could effectively guarantee the bandwidth fairness of TCP flows and is more robust than the existing methods.

References

Allman, M., 2000. A web server's view of transport layer. *ACM Communication Review*, **30**(5):10-20.

Chiu, D., Jain, R., 1989. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *ACM computer networks and ISDN systems*, **17**:1-14.

Feng, W.C., Shin, K.G., Kandlur, D.D., Debanjan, S., 2002. The blue active queue management algorithms. *IEEE/ACM Transactions on Networking*, **10**(4):513-528.

Golestani, S.J., Sabnani, K.K., 1999. Fundamental observations on multicast congestion control in the Internet. *IEEE In Proc. INFOCOM'99*, **1**:990-1000.

Habib, A., Bhargava, B., Fahmy, S., 2002. A Round Trip Time and Time-out Aware Traffic Conditioner for Differentiated Service Networks. IEEE In Proc. ICC'02, p.981-985.

Jeonghoon, M., Richard, J.L., Venkat, A., Jean, W., 1999. Analysis and comparison of TCP Reno and Vegas. *IEEE In Proc. INFOCOM'99*, **3**:1556-1563.

Mathis, M., Semke, J., Mahdavi, J., Ott, T., 1997. The macroscopic behavior of the TCP congestion avoidance algorithm. *ACM Computer Communication Review*, **27**(3):67-82.

Matsuda, T., Nagata, A., Yamamoto, M., 2002. TCP Rate Control Using Active ECN Mechanism with RTT-based Marking Probability. IEEE In Proc. CQR'02, p.112-116.

Nandy, B., Seddigh, N., Pieda, P., Ethridge, J., 2000. Intelligent Traffic Conditions for Assured Forwarding Based Differentiated Service Networks. IFIP. High Performance Networking, p.187-198.

Padhye, J., Firoiu,V., Towsley, D., 1998. Modeling TCP Throughput: A Simple Model and Its Empirical Validation. ACM In Proc SIGCOMM'88, p.303-314.

Seddigh, N., Nandy, B., Pieda, P., 1999. Bandwidth assurance issues for TCP flows in a differentiated service network. *IEEE GLOBECOM'99*, **3**:1792-1798.

Stoica, I., Shenker, S., Zhang, H., 1998. Core-stateless Fair Queuing: Achieving Approximately Fair Bandwidth Allocations in High-Speed Networks. ACM In Proc. SIGCOMM'98, p.118-130.