A NOVEL HIGH-PERFORMANCE TRANSPORT PROTOCOL CONSIDERING FAIRNESS WITH TCP IN LONG-DISTANCE HIGH-SPEED NETWORK

Fumiaki TAMESHIGE [*‡], *Ken-ichi BABA* [†‡], *Masaaki NORO* [‡], *Shinji SHIMOJO*[†‡]

[*] tamesige@ais.cmc.osaka-u.ac.jp, baba@cmc.osaka-u.ac.jp
[*] Graduate School of Information Science and Technology, Osaka University
[†] Cybermedia Center, Osaka University
[‡] JGNII Osaka Research Center, National Institute of Information and Communication Technology

ABSTRACT

Recently, high performance data transfer services is needed in long-distance high-speed networks. However, it has been reported that data transfer with TCP cannot achieve high throughput in LFN. As one of the solution overcome that, the data transfer protocol UDT which has rate control and adds reliability based on UDP is proposed. However the high volume data transfer with UDT is much aggressive to bottleneck bandwidth when multiple UDT flows coexist in the same path. Moreover, since high-speed data transport protocols influence coexistence of heterogeneous flows, we need to consider a new protocol and a new network control method. In this paper, we propose a novel control method to realize an effective high-speed data transfer. Our proposed method available in long-distance high-speed networks combines two approaches; UDT-g which can estimate appropriate bandwidth and RED-i which prevent continuous packet drop with instantaneous queue length. In conclusion, the simulation results show that our proposed method can improve loss probability with keeping high throughput.

1. INTRODUCTION

As network infrastructures supporting 1 Gbps and 10 Gbps links began emerging, new applicaions such as Grid emerge and require new control mechanism. Then the need for high performance data transfer services is becoming more and more critical in these applications. They need to transfer the distributed high volume data with high-speed and reliability. However TCP widely used on the Internet have notably poor efficiency and fairness in Long Fat pipe Network (LFN) which has large Bandwidth Delay Product (BDP) because of essential nature of TCP congestion control mechanism [1]. In AIMD (additive increase, multiplicative decrease) window flow control, TCP is designed to increase its sending rate by one segment per RTT when there is no congestion event. when there is a congestion event, the sending rate is decreased by half. That is, because its throughput is inverse proportional to RTT, TCP is not adequate in distributing data transport applications.

Recently, various new transport protocols have been proposed with the aim of efficient use of the abundant resources in LFN [2]. UDT (UDP based Data Transfer)[3] is especially noted compared with other protocols such as HighSpeed TCP in experiment results for substantial high throughput [4]. In the fact, UDT employs an AIMD rate control algorithm that takes a bandwidth estimation technique to determine the best increase parameter for efficiency.

On the other hand, it is important to consider not only efficiency, but also friendliness in LFN. In this paper, we define that the friendliness in LFN is the suppression of the influences with coexistence of heterogeneous flows, because the high-speed transport protocol can coexist with conventional traffic. For example, it is desirable to use excess bandwidth with high-speed data transfer without much influence to conventional traffic of applications such as Web, mail, streaming and so on in the general shared network. TCP becomes less optimal as network BDP increases, although it is still dominant in today's Internet. It is also important to consider the queuing delay in current Internet environments, because such applications are sensitive to communication delay and jitter,e.g. streaming applications. Consequently, it is important to use up wide bandwidth with suppressing the influence to other flows. In terms of friendliness, UDT can share bandwidth almost equally in small networks, but it is possible to take much more bandwidth than TCP in LFN [3] [4].

In this paper, we first demonstrated the problem when multiple UDT flow coexist with TCP flows over the same path. After that, we propose the combination method of UDT-g which improves estimating bandwidth and RED-i which prevent continuous packet drop with instantaneous queue length. In conclusion, we demonstrate the effectiveness of our proposed method.

2. OUTLINE OF UDT

2.1. Specification of UDT

UDT works over UDP with reliability and congestion control mechanisms. We show a concept of UDT in Fig. 1. If you want to know more detail, you can see [3].

1. Bandwidth Estimation Technique
   UDT uses a packet pair (PP) approach to estimate bandwidth. In every 16 packets, the sender does not have to wait until the next packet sending time, but send out two consecutive packets without inter-packet delay to form PP. The receiver, after receiving the PP, will calculate the link capacity from packet interval and send it back in the next ACK.
2. Rate Control and Flow Control
   UDT uses these two mechanisms: rate and window flow control. Rate control is the major mechanism used to tune the packet sending period, whereas window control is a supportive mechanism used to specify a dynamic threshold that limits the number of NAK packets. The rate control of UDT uses DAIMD algorithm which can realize high throughput even if in LFN [3]. the sum of SYN and RTT. UDT can realize a high performance transfer, considering efficiency and fairness and stability by these two controls.
3. Data Reliability
   Both peer sides maintain a loss list, which is used to record the sequence numbers of lost packets. The receiver sends ACK and NAK periodically to inform the sender of lost information. UDT can realize data reliabilty.

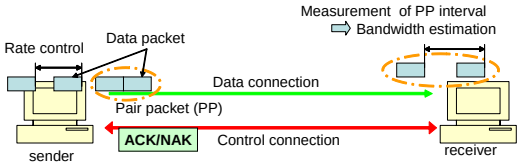

Fig. 1. Data transfer of UDT

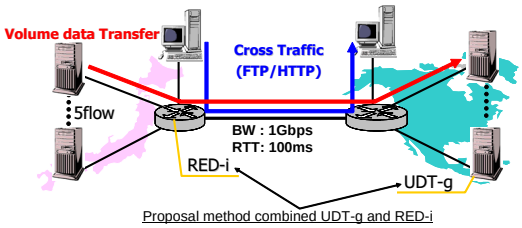

Fig. 2. The concept of our strategy and environment (including Simulation model)

2.2. Problem Description

UDT can realize high performance data transfer when one UDT flow exist in LFN. However, UDT generated a lot of packet loss when multiple flows coexist in the same path. Especially, this performance degradation can be seen in long-distance network. This is because each UDT flow estimates higher than actual bandwidth which is called fair shared bandwidth. In the case of one UDT flow, UDT can realize effective transfer even if in LFN because an estimate bandwidth is almost equal to bottleneck bandwidth.

In order to measure the bandwidth, the use of PP to measure available bandwidth has had more mixed results. Dovrolis, et al. points out that using PP leads to a value referred to as Asymptotic Dispersion Rate, which is a value between available bandwidth and bottleneck link capacity [5]. Additionally, DAIMD algorithm uses the difference between current sending rate and estimate bandwidth. UDT can increase sending rates in spite of congestion, if it doesn't estimate appropriate bandwidth value. Moreover, the degradation of a performance appears notably in LFN, because the estimate bandwidth depends on past value greatly and network state can not be reflected immediately.

3. PROPOSED METHOD

In this section, we propose two combination approaches to utilize efficiently high-speed network, when high volume data transfer flows compete TCP flows in LFN. We propose a novel data transfer protocol and active queue management by using the concept of UDT. One is UDT-g which improves the estimate bandwidth calculation and the other is RED-i which is RED queue mechanism based instantaneous queue length. Our concept is shown in Fig. 2

3.1. UDT-g (UDT with gentle bandwidth estimation)

To estimate bandwidth, UDT uses EWMA calculations with low constant smoothing value. However, each measurement by pair packets has substantial error and its values are totally high, because the network load is inevitably high in the high volume data transfer. If α is constantly at a low value, it is difficult to reflect the network state in a long-distance network because the estimate bandwidth depend on the past measurement value. In our proposed method, to reduce RTT influence, we proposed the following algorithm which control estimate bandwidth on purpose.

Proposal Algorithm

if ($ave\_bw \leq curr\_bw$)
    ave_bw = ave_bw * 0.875 + curr_bw * 0.125;
else
    $ave\_bw = ave\_bw * (1 - \alpha) + curr\_bw * \alpha$;

In this algorithm, we first compared maintaining average value of estimate bandwidth. If current measurement is higher than the average value, UDT-g uses a default smoothing value of 0.125. Otherwise, UDT-g uses α ($0.125 \leq \alpha \leq 1$) in its calculation of the estimate bandwidth. When multiple high volume data transfer flows coexist in the same path, UDT-g can reduce aggressive packet sending. Because the difference between current sending rates and the estimate bandwidth is smaller than conventional methods. UDT-g realize a stable DAIMD rate control. We believe that this α value is the trade-off between efficency and fairness. Here we use alpha value 1. UDT-g uses the last measurement value, when network is congested.

3.2. RED-i (RED based instantaneous queue length)

FIFO queuing with Drop-tail (DT) is the most widely implemented and deployed mechanism in today's routers because of its simplicity. However, it has been reported that DT queue has some existing problems, for example global synchronization and consecutive packet loss. RED (Random Early Detection) hopes to eliminate the bias against bursty traffic by dropping before the buffer over flow [6]. RED estimates the congestion level by monitoring and updating its average queue size. RED compute the average queue size by EWMA based on following equation.

$$Q_{ave} = q_w * Q_{cur} + (1 - q_w) * Q_{ave} \quad (1)$$

Fig. 3 shows the dropping profile of RED. Fig. 4 shows the relationship between the average queue length and the instantaneous queue length. RED uses a low-pass filter to calculate the average queue size. Thus, the short-term increases in the queue size as a result of the bursty traffic from transient congestion and does not result in a significant increase in the average queue size.

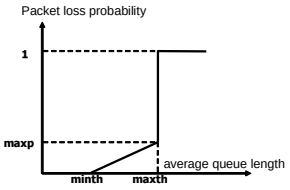

Fig. 3. RED mechanism

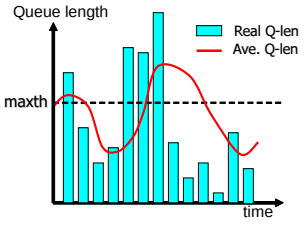

Fig. 4. The changes of queue length

However, because the q_w often takes a much smaller value, the average queue length is much different from the instantaneous queue length in LFN. There are various problems [7]. For example when queue size increases or decreases drastically, a lot of packets might drop unnecessarily, because the average queue size can not keep up with the instantaneous queue size. We can especially take into consideration that performance decreases notably in our intended model like LFN, because it is inevitable to delay the signal of the congestion in long-distance networks. In this paper, we defined conventional method as RED-a (RED based average queue length) and proposed method as RED-i (RED based instantaneous queue length).

4. PERFORMANCE EVALUATION

In order to evaluate the performance of our proposed method, we have simulated it by NS2. Also we implemented our method to experimental environment utilizing JGN2, the test-bed network in

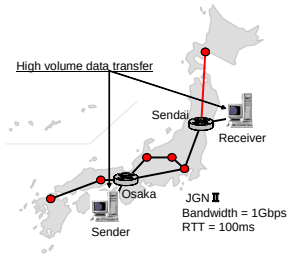

Fig. 5. Experimental setup in JGN2

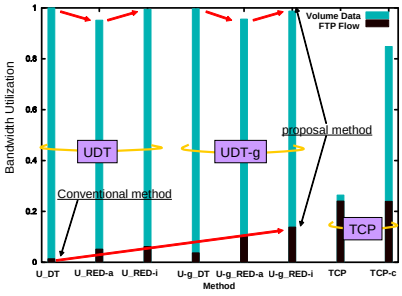

Fig. 6. Property of Throughput (FTP and volume data transfer model)

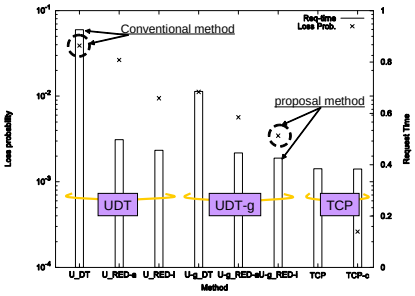

Fig. 7. Request time and packet loss probability (HTTP and volume data flow model)

Japan, provided by NiCT. Our simulation model is shown in Fig. 2 and our experimental model is in Fig. 5. In this paper we report simulation result only when multiple volume data flows compete with TCP in LFN because of lack of space. We assume two TCP traffic models: FTP and HTTP.

4.1. Simulation Model

We assume the environment of volume data transfer with UDT in Fig. 2. We assume a small number peer side to transfer volume data. We use UDTv2 which is described in [3] as protocol of normal volume data transfer. We use TCP NewReno which is the main current version in today's Internet. All packet sizes are 1500byte. RED parameters are set as follows: maxp =0.1, minth =BDP/2, maxth = BDP. TCP use 64 Kbytes of window size which is default size in Linux. On the other hand, TCP-c means that window size of TCP is adjusted to the BDP by using the window scale option. When we use UDT, we have evaluated DT, RED-a and RED-i as a queue management. However, when we used TCP as high volume data transfer, we used RED queue because it is not much difference between the performance of DT and RED.

4.2. Performance Analysis

We have evaluated the performance when 5 volume data flows compete with 50 FTP flows in Fig. 2. In this evaluation, FTP with TCP limits its windw size 64Kbytes. Fig. 6 shows that the performance of throughput. From Fig. 6, the bulk data transfer with TCP can not realize high throughput even if scaling TCP window size. On the other hand, UDT can use up the bottleneck bandwidth at the cost of FTP performance. Furthermore UDT-g can realize high throughput without greate influence to FTP flows becase UDT-g can estimate apropriate bandwidth. From this, we consider that UDT is too aggressive to FTP flows in LFN. In Fig. 6, UDT-g with DT may be unfriendly, but we believe that UDT-g can be friendly wrt to TCP if queue size is smaller. UDT with RED-a can not use up bottleneck bandwidth because of unnecessary packet losses. Especially, UDT-g with RED-i queue can realize data transfer effectively.

Next, in the future, we assume that the bulk data transfer will compete with the short lived TCP like Web in Fig. 2. The size of the document which a server transmits follows the pareto distribution with an average size of 10Kbyte and a shape parameter is 1.3, and an interval between each clients' request to a server for document transmission follows the exponential distribution with an average of 800ms. we assume that the HTTP traffic is about 20 percent of the bottleneck link. Fig. 7 shows the transmission time of one document and packet loss probability. We omit the figure of the property of throughput because the performance of throughput is not so much different from Fig. 6. In Fig. 7, the transmission time with DT is twice as with RED, because UDT with DT queue has the characteristic which keep higher occupation in buffer of queue. In addition, most HTTP flows might finish within the slow start phase in the TCP algorithm, because the average file size is set at 10Kbyte in our simulation. One packet loss in TCP will greatly impact to the transmission time. At the result, UDT flows with DT queue greatly affect HTTP traffic. Furthermore, the transmission time of the Web document will improve with the RED queue, especially if we use the combined approach of UDT-g and RED-i, which can reduce the burst packet loss and prevent the increase of queuing delay. We expect that UDT-g can greatly improve performance in actual experiment environments since reducing packet losses can improve the CPU load. From these, our proposed method realizes high-speed data transfer by UDT without impactacting the TCP flows.

5. CONCLUSION

We have reported high volume data flow with UDT is much aggressive to TCP when multiple UDT flows coexist in the same path. we proposed a new control for high performance data transfer. UDT-g achieves gentle increment of the rate control in long-distance network. RED-i can control queue length and improve loss probability. We showed through simulations that the proposed approach outperforms the conventional methods. Future work will include further investigation of effective high volume data transfer and active queue management control which is much proper to LFN.

6. REFERENCES

[1] S. Floyd, "Highspeed TCP for large congestion window," RFC 3646, IETF, Dec. 2003.

[2] R. L. Cottrell, S. Ansari, P. Khandpur, R. Gupta, F. Hughes Jones, M. Chen, L. MacIntosh, and F. Leers, "Characterization and evaluation of TCP and UDP-based transport on real networks," in *3rd Workshop on Protocols for Fast Long-distance Networks*, Feb. 2005.

[3] Y. Gu, X. Hong, and R. Grossman, "Experiences in the design and implementation of a high performance transport protocol," in *SC2004*, Nov. 2004.

[4] K. Kumazoe, Y. Hori, M. Tsuru, and Y. Oie, "Transport protocols for fast long-distance networks : Evaluation of their penetraion and robustness on JGN2," in *PFLDnet*, Feb. 2005.

[5] C. Dovrolis and D. Moore, "What do packet dispersion techniques measure?," in *INFOCOM2001*, pp. 905–914, Apr. 2001.

[6] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397–413, Aug. 1993.

[7] K. Yasukawa, K. Baba, and K. Yamaoka, "Drop precedence mapping on dynamic class assignment method," *The Mediterranean Journal of Computers and Networks*, vol. 1, no. 1, pp. 37–46, July 2005.