*Journal of Electrical Engineering*
*The Institution of Engineers, Bangladesh*
*Vol. EE 31, No. I & II, December 2004*

# TCP Friendly Rate Control Protocol Performance with Many Flows

**Mohammad Nazmul Ahsan[1] and Md. Saifur Rahman[2]**

[1]Department of Computer Science & Engineering, DUET, Gazipur
[2]Department of Electrical & Electronic Engineering, BUET, Dhaka
Email: nahsan06@yahoo.com

**ABSTRACT**

**The TCP-Friendly Rate Control Protocol (TFRC), is a rate-based end-to-end congestion control protocol. TFRC uses a model for steady state Transmission Control Protocol (TCP) throughput to limit the sending rate and assure fair behaviour against competing flows. Instead of reacting to single congestion events (in the form of packet loss) like TCP, the TFRC protocol changes its sending rate in response to the loss rate, sampled over a certain amount of time. While TFRC achieves the same long-term throughput as a conformant TCP flow, its short-term sending rate is more stable. This makes the protocol suitable for traffic where sudden rate changes are undesirable (e.g. video or audio streams). In the last several years extensive research have been done on TCP-Friendly Protocols and it shows that TFRC is able to share a bottleneck or congested router's bandwidth fairly with competing TCP flows. But all the research so far done uses a simulated test-bed or real life internet where number of flows are limited. In this paper we show the performance of TFRC with a large number of flows and found some discrepancies with previous research outcome. It is shown that TFRC is a scalable protocol though at very high loss rate, its fairness performance degrades.**

***Keywords* :** TFRC, TCP, UDP, NS, equivalence ratio, coefficient of variation, drop tail queuing.

## 1. INTRODUCTION

TCP is the dominant transport protocol in the Internet and 95% of the Internet traffic is carried by TCP. The basic algorithm incorporated by TCP is Congestion Avoidance and Control [1] which uses an Additive Increase Multiplicative Decrease (AIMD) algorithm. TCP connection probes for available network bandwidth by slowly (additively) increasing a congestion window (sending rate is controlled by this congestion window) and on detecting a congestion (indicated by the loss of one or more packets), reducing the congestion window multiplicatively by a factor of two. This rapid back off in response to congestion is the key to TCP's success in avoiding congestive collapse in the Internet. However, this algorithm makes two problems with TCP. First, it makes TCP extremely susceptible to bandwidth stealing by other transport protocols which do not implement any congestion control techniques. Second, this rapid reduction of sending rate in response to congestion makes TCP unsuitable for real time streaming applications where smooth sending rate is extremely necessary. That is why these applications use User Datagram Protocol (UDP). The widespread development of these applications poses two major threats to the Internet. First, these applications could contribute to a new congestive collapse of the Internet. Second, these applications will consume an unfairly large portion of resources when run side-by-side with TCP applications. Extreme research has been going on for the last several years to develop a new transport protocol which will be responsive to congestion, suitable for real time streaming application and will fairly share resources with coexisting TCP applications. Generically these protocols are termed as TCP Friendly Protocol. TCP Friendly Protocol are of two types. Some are window based and some are equation based. TFRC [2] is an equation based TCP Friendly transport protocol. It was developed at AT&T Center for Internet Research by S. Floyd. In this paper the steady state behaviour of TFRC with a large number of flows is studied.

## 2. BACKGROUND AND RELATED WORKS

As the use of multimedia streaming application becomes widespread among internet community, TCP Friendly Transport Protocol is the major concern for present internet researchers. Sisalem and Schulzrinne [3] proposed a Loss–Delay based adjustment algorithm to adapt transmission rate for multimedia application in a TCP Friendly way. It relies on Real Time Protocol (RTP) for periodic feedback reports from the receiver to estimate loss rate and round-trip times. The Rate Adaptation Protocol (RAP) [4] is an TCP Friendly adaptation scheme that uses an AIMD rate control scheme based on regular acknowledgements sent by the receiver which the sender uses to detect lost packets and estimate the Round Trip Time (RTT). Bansal and Balakrinshna in [5] consider binomial congestion control algorithms, where a binomial algorithm uses a decrease in response to a loss event that is proportional to a power *l* of the current window, and otherwise uses an increase that is inversely proportional to the power of *k* of the current window. AIMD congestion control is special case of binomial congestion control that uses $l = 1$ and $k = 0$. Authors in [5] consider several binomial congestion control algorithms that are TCP compatible and that avoid TCP's drastic reduction of the congestion window in response to a loss event. In the TEAR protocol (TCP Emulation at the Receivers) [6], the receiver emulates the congestion window modifications of a TCP sender, but then makes a translation from a window-based to a rate-based congestion control mechanism. The receiver maintains an exponentially weighted moving average of the congestion window, and divides this by the estimated round-trip time to obtain a TCP-friendly sending rate. The TCP-Friendly Rate Control Protocol (TFRCP) [7] uses an equation based congestion control mechanism for unicast traffic where the receiver acknowledges each packet. At fixed time intervals, the sender computes the loss rate observed during the previous interval and updates the sending rate using the TCP response function described in [8]. Since the protocol adjusts its sending rate only at fixed time intervals, the transient response of the protocol is poor. Most comprehensive work on TCP Friendly Transport Protocol was done in [2] and protocol specification is described in [9]. But performance of TFRC was observed in [2] with only 32 flows in low loss rate. Recently TFRC protocol is modified in [10] for wireless network.

## 3. TCP FRIENDLY RATE CONTROL PROTOCOL

TFRC is a sender-based scheme that works by continuously adjusting the source's sending rate based on Eq. 1 [2]. The sender is to adjust its transmission rate based on observed loss event rate and RTT. The adjustment of sending rate to achieve TCP-friendliness is based upon the TCP throughput model described in [8]. The sender uses the slow start technique at the beginning of transmission phase during which it tries to multiplicatively increase its sending rate at every RTT until it detects a loss. Packet losses are identified by gaps in the sequence number of the transmitted packet at the receiver module, as in RAP. The receiver sends feedback to the sender at regular intervals. The sender then adjusts the rate accordingly based on this feedback. When the receiver notices the loss, it immediately notifies the sender and the sender adjusts its sending rate based on the TFRC Eq. 1.

$$T = \frac{s}{R\sqrt{\frac{2p}{3}} + t_{RTO}(3\sqrt{\frac{3p}{8}})p(1+32p^2)} . \qquad (1)$$

This gives an upper bound on the sending rate $T$ in bytes/sec, as a function of the packet size $s$, round-trip time $R$, steady-state loss event rate $p$ and the TCP retransmit timeout value $t_{RTO}$.

## 4. EXPERIMANTAL EVALUATION

Main objective of this experiment is to observe the performance of TFRC in terms of *Equivalence Ratio* and *Coefficient of Variation* towards the competing TCP traffic flows when such number of flows is very high. In other words we want to see the scalability issue of TFRC. Let us first define the above mentioned performance metrics: *Equivalence Ratio* and *Coefficient of Variation.*

*Coefficient of Variation* (CoV) is a measure of variability of throughput over a particular time interval. Throughput $R_{\delta, F}(t)$ of a given data flow $F$ using $s$ byte packets at time t, measured at a time interval $\delta$ can be defined as:

$$R_{\delta,F}(t) = \frac{s \times p}{\delta}, \qquad (2)$$

where $p =$ No. of packet sent by $F$ between $t$ and $t + \delta$.

Throughput of the flow between time $t_0$ and $t_1$, where $t_1 = t_0 + n\delta$ , can be expressed by the time series : $\{R_{\delta,\ F}\ (\ t_0 + i^*\ \delta)\}_{i\ =\ (0\ -\ n)}$ . *Coefficient of Variation* is standard deviation divided by the mean of this time series; a lower value implies a flow with a range of throughput more closely clustered around the average.

To compare the throughput of two flows a and b at a given time interval $\delta$, *equivalence* $e_{\delta,a,b}(t)$ at time t is defined in [2] as :

$$e_{\delta,a,b}(t) = \min\left(\frac{R_{\delta,\mathrm{a}}(t)}{R_{\delta,\mathrm{b}}(t)}, \frac{R_{\delta,\mathrm{b}}(t)}{R_{\delta,\mathrm{a}}(t)}\right), \qquad (3)$$

for $R_{\delta,\mathrm{a}}(t) > 0$ or $R_{\delta,\mathrm{b}}(t) > 0$.

Taking the minimum of the two ratios ensures that the resulting value remains between 0 and 1.

Equivalence of two flows between time $t_0$ and $t_1$, where $t_1 = t_0 + n\delta$, is characterized by the time series:

$$\{e_{\delta,\ a,b}(\ t_0 + i^*\ \delta)\}_{i = (0 - n)}\ .$$

The average value of the defined elements of this time series is called the *Equivalence Ratio* of the two flows at time interval δ. The closer it is to 1, the more "equivalent" the two flows are. In other words if *Equivalence Ratio* is 1, two flows will share the bottleneck bandwidth fairly. In our case both TCP and TFRC flows will obtain 50 % of the bottleneck bandwidth. In Ref. [7] we find another performance metric, *Friendliness Ratio* defined by

$$F_r = \frac{\sum T_C}{\sum T_T}. \qquad (4)$$

Where $T_C$ is average throughput of TCP Friendly flows and $T_T$ is the average throughput of TCP flows. But *Equivalence Ratio* gives a better visual representation of the closeness of the throughput achieved by the two protocols.

## 5. SIMULATION SCENARIO

We have used the simulator ns-2 [11] from the VINT project at U.C. Berkeley/LBNL to perform our experiment. The simulator is event driven and runs in a non-real-time fashion. In ns-2 user can simulate nodes, routers, links and shared media. A rich set of protocol objects can be attached to nodes, usually as agents. User may choose between various types of applications. In this simulation, we used the FTP application for the TCP agent and an application using a constant bit rate (CBR) traffic pattern for TFRC agent. We used SACK TCP. We configure the router using DropTail (FIFO) queuing.

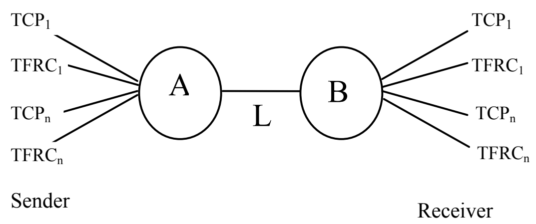


**Fig. 1:** Simulation topology.

Figure 1 illustrates the simulation topology. In each simulation, equal number of TCP and TFRC senders transmits through a shared bottleneck to an equal number of receivers. Congestion will occur in router A at the input to link L. Each sender has an unlimited amount of data to send.

## 6. SIMULATION PARAMETERS

A fair comparison can only be achieved with careful selection of simulation parameters. In our simulation topology, the intermediated routers are connected by a bottleneck link with bandwidth set to 4 Mbps and link delay of 20 ms. Traffic from all sources are sharing this bottleneck bandwidth. The side links connected to the bottleneck link have bandwidth of 1 Mbps. The routers have a single output queue for all attach link and DropTail scheduling discipline is used. All simulations were run long enough until they achieved steady state behavior. Table 1 summarizes important simulation parameters used in these experiments.

**Table 1 :** Simulation parameter.

| Parameter | Default Value |
|---|---|
| Packet Size | 1000 Bytes |
| Bottleneck Bandwidth | 4Mbps |
| Bottleneck Delay | 20ms |
| Side link Bandwidth | 1Mbps |
| ACK Size | 40Bytes |
| Simulation Time | 200s |

## 7. EXPERIMENTAL RESULT

We have conducted several simulations with increasing number of flows. First five simulations are done with 40, 60, 100, 160, 200 flows. In every simulation, 50 % of the flows are TCP and 50 % of them are TFRC. Equivalence Ratio for TCP vs TFRC is determined by comparing the average throughput of all TCP flows with that of TFRC flows in 10 second interval. Equivalence Ratio for TFRC vs TFRC is determined by comparing the average throughput of half of the TFRC flows with that of the rest of the TFRC flows. Equivalence Ratio for TCP vs TCP is also determined in the same fashion.

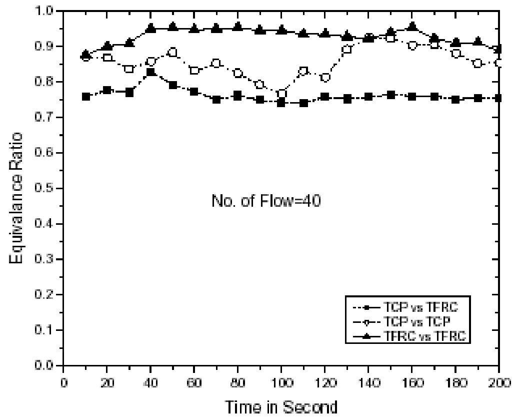


**Fig. 2:** Equivalence ratio for 40 flows.

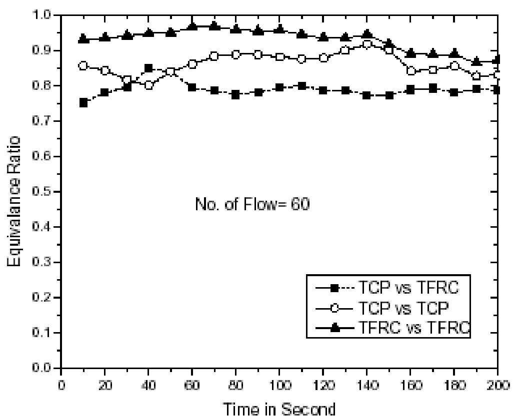


**Fig. 3:** Equivalence ratio for 60 flows.

As *Equivalence Ratio (ER)* indicates how competing flows share the bottleneck bandwidth, 0.8 and above can be considered as a very good figure. At *ER* = 0.8, one flow gets 55% share and other gets 45% share of the total bandwidth. From the Fig. 2 to Fig. 6, we find that bandwidth is shared fairly by TCP and TFRC flows at steady state condition. TCP flows also share the bandwidth among them fairly as indicated by the Equivalence Ratio for TCP vs TCP. Same decision can be taken for TFRC flows from *ER* for TFRC vs TFRC. When number of flows become significant, steady state condition reaches with a delay as indicated by Fig. 5 and Fig. 6.

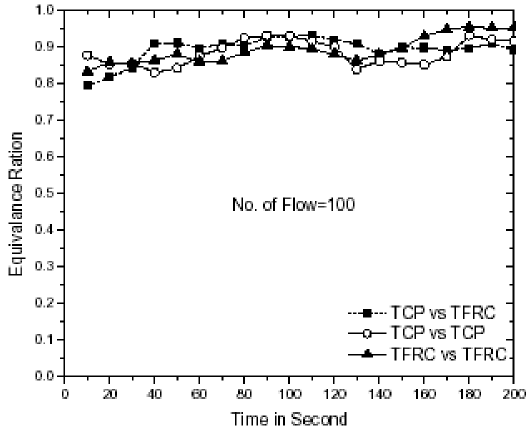


**Fig. 4:** Equivalence ratio for 100 flows.

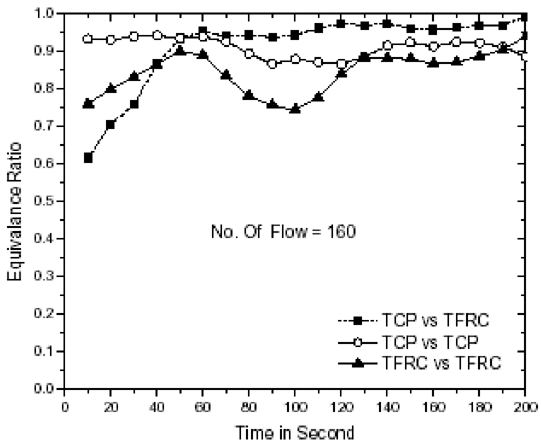


**Fig. 5:** Equivalence ratio for 160 flows.

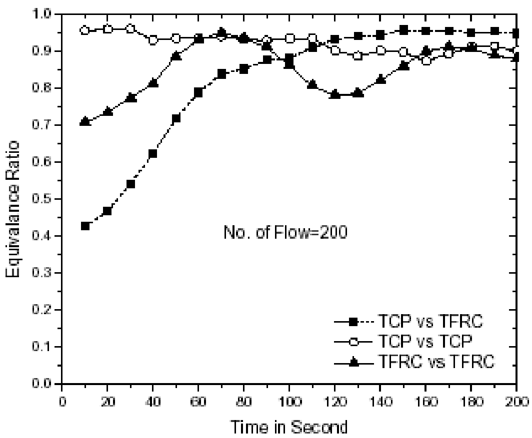


**Fig. 6:** Equivalence ratio for 200 flows.

Let us now focus on the packet loss rate. From Fig.7, we find that congested router A drops packet fairly among the protocols. One thing becomes obvious that TCP and TFRC remain friendly with each other at loss rate as high as 15 to 16 percent.

Figure 8 shows that CoV of TFRC is less than CoV of TCP flows. So throughput of TFRC is less variable than TCP even in high loss rate. This makes TFRC more suitable than TCP for real time streaming application.

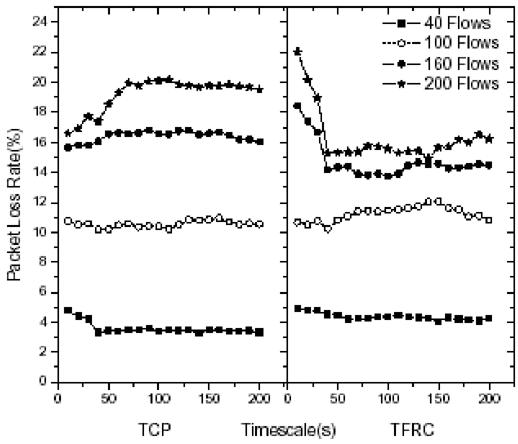


**Fig. 7:** Packet loss ratio for different no. of flows.

Next three simulations are done with 400, 600 and 800 flows. Figure 9 shows that bandwidth is not shared fairly when the number of flows become very high. TCP becomes greedy with the bandwidth. Loss rate of TFRC also increases over TCP with increased number of flows though we have not shown it here. From Fig. 10 we find that CoV of TFRC is not less than TCP when the number of competing flows are very high. In Ref. [2], authors concluded that TFRC is scalable protocol without testing it with many flows.

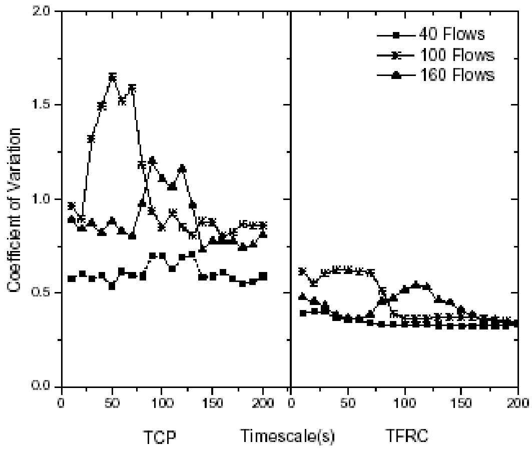


**Fig. 8:** CoV for different no. of flows.

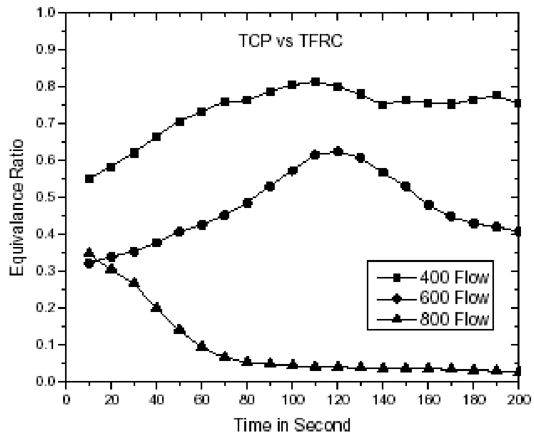


**Fig. 9:** Equivalence ratio with many flows.

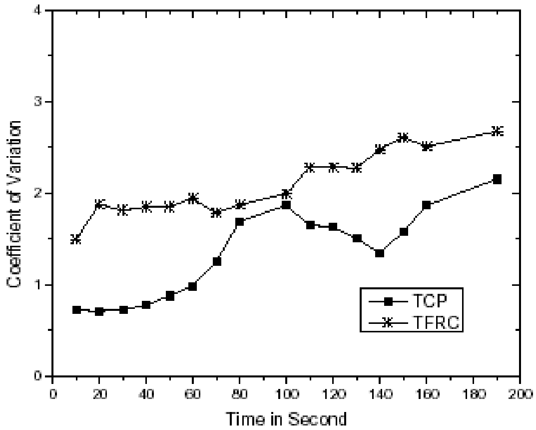


**Fig. 10:** CoV for 600 flows.

## 8. CONCLUSION

TCP is a much studied phenomenon. Same sort of experiment was done for TCP in Ref. [12]. But TFRC protocol is in rudimentary phase. There are many situations in which TFRC is net yet tested. In this paper, we only considered the steady state behavior of TFRC neglecting the transient behavior. Our verdict is that TFRC is a scalable protocol, but when number of competing flows increases and loss rate becomes significant, its fairness performance degrades.

## REFERENCES

[1] V. Jacobson, "Congestion Avoidance and Control," *in Proc. SIGCOMM Symposium on Communications Architectures and Protocols,* pages 314 – 329, August 1988.

[2] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-Based Congestion Control for Unicast Applications," in *Proc. SIGCOMM 2000*, August 2000.

[3] D. Sisalem and H. Schulzrinne, "The Loss-Delay Based Adjustment Algorithm: A TCP Friendly Adaptation Scheme," in *Proc. NOSSDAV98* ,1998.

[4] R. Rejaie, M. Handley, and D. Estrin, "An End to End Rate–based Congestion Control Mechanism for Realtime Streams in the Internet," in *Proc. INFOCOMM'99,* 1999.

[5] D. Bansal and H. Balakrishnan, "TCP- Friendly Congestion Control for Real-time Streaming Applications," *MIT Technical Report MIT-LCS-TR-806,* May 2000.

[6] I. Rhee, V. Ozdemir, and Y. Yi, "TEAR: TCP Emulation at Receivers – Flow Control for Multimedia streaming," *NCSU Technical Report,* April,2000.

[7] J. Padhye, J. Kurose, D. Towsley, and R. Koodli, "A Model Based TCP-Friendly Rate Control Protocol*," in Proc of NOSSDAV' 99*, 1999.

[8] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP Throughput: A Simple Model and its Empirical Validation," in *Proc SIGCOMM Symposium on Communications Architectures and Protocols,* August. 1998.

[9] S. Floyd, M. Handley, J. Padhye, and J. Widmer "TCP Friendly Rate Control: Protocol Specification," *RFC-3448,* January 2003.

[10] Mingzhe Li, Choong-Soo Lee, Emmanuel Agu, Mark Claypool, and Robert Kinicki, "Performance Enhancement of TFRC in Wireless Ad Hoc Networks," in *Proc. of the 10th International Conference on Distributed Multimedia Systems (DMS),* San Francisco, California, USA, September. 2004.

[11] Steve McCanne and S. Floyd (2003, August). The ns-2 Network Simulator [Online]. Available http://www-mash.cs.berkeley.edu/ns/.

[12] Robert Morris, " TCP Behavior with Many flows," in *Proc. of IEEE International Conference on Network Protocols,* Atlanta, Georgia, Oct. 1997.