future internet

MDPI

Article

Efficient Traffic Engineering Strategies for Improving the Performance of TCP Friendly Rate Control Protocol

Nalavala Ramanjaneya Reddy [1,*], Pakanati Chenna Reddy [1] and Mokkala Padmavathamma [2]

[1] Department of Computer Science and Engineering, Jawaharlal Nehru Technological University, Anantapur 515001, AndhraPradesh, India; chennareddy.cse@jntua.ac.in

[2] Department of Computer Science, Sri Venkateswara University, Tirupati 517502, Andhra Pradesh, India; mpadma@svuniversity.edu.in

* Correspondence: ramanji.nalavala@gmail.com; Tel.: +91-9989589227

Received: 31 August 2017; Accepted: 7 October 2017; Published: 1 November 2017

Abstract: Multimedia services will play a prominent role in the next generation of internet. With increasing real time requirements, internet technology has to provide Quality of Service (QoS) for various kinds of real time streaming services. When the bandwidth required exceeds the available network resources, network paths can get congested, which results in a delay in packet delivery and packet loss. This situation leads to the design of new strategies for congestion avoidance and control. One of the popular and appropriate congestion control mechanisms that is useful in transmitting multimedia applications in the transport layer is TCP Friendly Rate Control Protocol (TFRC). However, TFRC still suffers from packet loss and delay due to long distance heavy traffic and network fluctuations. This paper introduces a number of key concerns like enhanced Round Trip Time (RTT) and Retransmission Time Out (RTO) calculations, Enhanced Average Loss Interval (ALI) methods and improved Time to Live (TTL) features are applied to TFRC to enhance the performance of TFRC over wired networks.

Keywords: congestion control; TCP; RTT; RTO; TFRC

1. Introduction

A computer network is a collection of computers that are interconnected together for the purpose of resource sharing. With increasing real-time requirements, a computer network has to provide different types of applications and services. Today, billions of computers are connected to the internet and thousands of terabytes of data are being transmitted per hour.

When too many packets arrive at one place, the network will not be able to handle the packets; this situation leads to congestion in the network. Congestion may occur when the load on the network is greater than the capacity of the network. At the point when the network is congested, the performance of the network may deteriorate or become worse in the realm of delay, packet loss, etc. Because of the explosive growth of the internet, congestion avoidance and control play an important role in transport layer functionality [1]. The objective of congestion control is to maximize the network performance in all circumstances.

Transmission Control Protocol (TCP) is one of the transport layer protocols used for bulk data transfer. However, TCP is not always quite appropriate for transmitting real time multimedia applications like online video streaming applications due to the manner it handles the congestion [2]. On the internet, the majority of internet traffic is TCP based [3], so end systems should have to react to congestion over the network in order to achieve high network utilization and a fair share of bandwidth. TCP New Reno [4] is a variant of TCP, which is employed to regulate the congestion by adjusting the

number of data packets that can be transmitted without affirmation. Moreover, once packet loss is identified in the network, TCPNew Reno reduces its congestion window to manage congestion.

This accelerates to insufficient data transmission for continuous play back videos. To avoid the potential impendences to the internet, TFRC [5] is designed. The main objective of TFRC is to provide smoother throughput than TCP for real time applications and its working functionality can be represented by Figure 1.

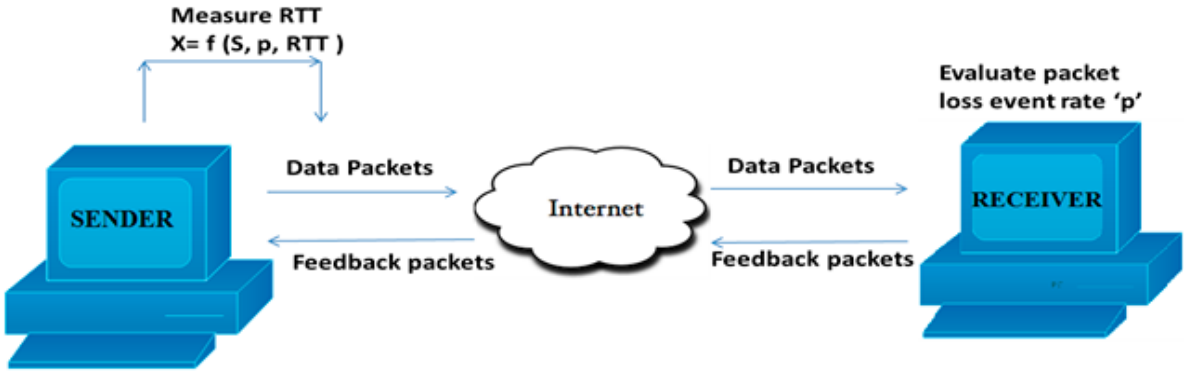

Figure 1. Functionality of TCP Friendly Rate Control mechanism.

TFRC is an end to end transport layer congestion control mechanism for unicast streams in the best effort surroundings. TFRC can react to changes in transmission capacity and it can alter its sending rate based on round trip time and loss event rate. TFRC alters its sending rate depending on the TCP Reno equation as given below

$$X = \frac{S}{RTT\sqrt{\frac{2bp}{3}} + 3 * RTO\sqrt{\frac{3bp}{8}}p(1 + 32p^2)} \quad (1)$$

where, X represents throughput in bytes per second, S denotes the packet length in bytes, RTT is round trip time, b indicates the number of packets that are acknowledged by one Acknowledgement (ACK), the loss event rate is denoted by 'p' and RTO represents retransmission time out value. This condition keeps up an unfaltering state sending rate to the network to avoid unexpected vacillations. By setting the parameter acknowledged by one ACK to 1 and by considering four times of round trip time as one retransmission time out value, TFRC throughput equation can be framed in terms of packet length, loss event rate and round trip time as follows

$$X = \frac{S}{RTT\sqrt{\frac{2p}{3}} + 12 * RTT\sqrt{\frac{3p}{8}}p(1 + 32p^2)} \quad (2)$$

In view of the above condition, TFRC works as follows. At the initial step, the receiver measures loss event rate by using Average Loss Interval method and forwards this information to the sender. Sender calculates round trip time based on received information. Sender calculates its sending rate as a function of loss event rate and round trip time and this can be represented as f(S,p,RTT). The sender keeps up its sending rate in view of its computed rate from the above condition. Finally, TFRC sending rate is primarily a function of loss event rate and round trip time.

TCP Friendly Rate Control protocol is primarily designed to present the smooth and best possible service for streaming applications which is supporting it best in the internet environment. TRFC performance is highly affected by throughput and packet loss due to unfair RTT and RTO calculations. Further TFRC requires a sophisticated mechanism for calculating accurate pocket loss rate. To achieve Quality of Service for transferring online multimedia streaming applications, more sophisticated mechanisms are needed for TFRC. This encourages the researchers to propose

various strategies like Tuning RTT and RTO dynamically by calculating them using a better approach, introducing mean and confidence interval into average loss interval method and modification to TTL computation method to reduce packet dropping using RSA to prevent TTL modification to improve the performance of TFRC in terms of throughput, delay and packet loss.

This paper is organized as follows. Section 2 describes related work done so far on TFRC. Section 3 presents the various proposed strategies for enhancing the performance of TFRC. In Section 4. Simulation environment is described. Metrics, simulation results and discussions of using the proposed technique to improve the throughput of TFRC are included in Section 5. Section 6 presents the conclusion.

2. Literature Survey

Numerous researchers have done some work to analyze and enhance TFRC. They have used simulations as well as real time test beds. A few of the related results are as follows. In [6], authors identified that TFRC suffers from transmitting multimedia applications over the internet. To increase the performance of TCP Friendly Rate Control, authors proposed an enhanced TFRC which substitutes TCP throughput equation recommended for the TFRC with a linear throughput equation. This linear equation avoids the collapse of the TCP connections and allows the TCP to utilize available bandwidth. This enhanced TFRC avoids congestion more aggressively. Results show that the proposed linear throughput equation gives more throughput than the classical throughput equation and it is more suitable for transmitting multimedia applications.

In [7], authors identified that TFRC supports fixed packet size flows and is not suitable for some continuous media applications. For that, authors proposed an enhanced TFRC to support variable size packet flows. They modified numerator of TFRC throughput equation to be the Maximum Transmission Unit (MTU) of the path, so it is suitable for variable packet flows. Modified TFRC behaves appropriately in response to congestion for various scenarios. Simulation results show that it is a superior TFRC protocol that has a wider range of viable packages and therefore more chances of being widely followed on the internet.

In [8], authors analyzed TFRC performance over wired and wireless networks, and concluded that TFRC suffers from significant performance degradation in terms of throughput. To enhance the performance of TFRC, authors proposed a new enhanced scheme called TFRC Veno. They replaced Reno equation with Veno equation called TFRC Veno. It is also a kind of enhancement which modifies TFRC at the sender side. TFRC Veno equation is derived from TCP Veno. With this addition of veno equation to TFRC, it results in better throughput and good fairness along with reduced packet loss over wireless networks. Simulation results show that, TFRC Veno improves TFRC performance over wireless networks significantly for multimedia transmission. When compared to Multiple TFRC (MULTFRC), the proposed method is friendlier to other network traffic and it achieves good throughput.

In [9] authors analyzed that TFRC protocol uses a slow start algorithm which leads to degrading network performance. To adapt network conditions dynamically, authors proposed a variant of slow start algorithm called Bandwidth Estimation-start (BE-Start) method to improve the performance of TFRC protocol. Simulation results show that with the inclusion of BE-Start algorithm to TFRC, throughput is effectively increased and packet loss is reduced. In [10] authors examined the behavior of TFRC-SP (Small Packets) over live streaming data like audio, VOIP, etc. They identified that TFRC-SP uses a slow start algorithm which is inappropriate for real time applications. To overcome this drawback, authors proposed a quick start algorithm to TFRC-SP, which increases sending rate and reduces unnecessary retransmissions. In [11] authors investigated that TFRC suffers from inaccurate sending rate adjustment in multi hop wireless adhoc networks. To overcome this drawback, authors proposed a variant of TFRC called RE-TFRC (Rate Estimation TFRC) to enhance the performance of TFRC in a wireless adhoc network environment. Result show that reduced round-trip time and better throughput than TFRC.

In [12], authors identified that conservativeness causes lower throughput and slower response time for TFRC. To eliminate the problems of TFRC, they proposed Adaptive TFRC (ATFRC) for faster responsiveness and convergence time with less conservativeness than TFRC when bandwidth is available. In [13], authors proposed a non-linear min-max averaging operation for finding packet loss event rate using average loss interval method. In [14], authors included min-max averaging operations in basic average loss interval algorithm, which helps to reduce the variance of the Average Loss interval (ALI) estimation by suppressing the correlation of neighboring loss intervals arising from the Additive Increase and Multiple Decrease (TCP-AIMD) behavior.

In [15], authors modified some fixed parameters of TFRC throughput equation, which results in increased throughput and less packet loss. In [16] authors proposed an enhanced queue management scheme for TFRC to reduce packet loss and to increase the throughput of bursty traffic. In [17], the authors proposed an enhancement to TFRC that uses a self-clocking mechanism to eliminate packet loss during bandwidth decreased handover. In [18], the authors proposed two strategies called bandwidth estimation method and new slow start mechanism to enhance the performance of TFRC in the view of bandwidth allocation among multiple flows and packet losses caused by over shooting.

In [19], the authors analyzed the performance of TFRC to transport multimedia applications over homogeneous wired networks and found that TFRC needs improvement in homogeneous wired networks. To increase the ability of TFRC, authors proposed State-TFRC (S-TFRC), which includes a feedback mechanism which exchanges link information and transmission parameters. Also, authors analyzed a Markov model to demonstrate S-TFRC protocol behavior in different events of congestion like timeouts, loss rates, and no-feedback timer expiration. NS-2.34 is used to test the environment of proposed S-TFRC and classical TFRC. Simulation results show that proposed S-TFRC produces better performance in terms of throughput, delay and transmission losses for real time applications.

In [20], the authors proposed a new receiver based retransmission scheme for TFRC to improve the effective retransmission ratio according to the forward path delay variance and also reduces unnecessary retransmissions. In [21], the authors suggested a fast startup scheme and new bandwidth estimation methods to overcome the problems of TFRC RTT-unfairness problem and overshooting problem. In [22], an enhanced TFRC control algorithm called Network State Control (NSTC) was presented, which introduces available bandwidth variation as network identification and limits sending rate in different network states. This scheme enhances TFRC performance in terms of decreased packet loss and comprehensively improves QoS and friendliness. In [23], authors proposed a new congestion control framework called Ultra-driven TFRC (UTFRC), which is more suitable for multimedia streaming applications than classical TFRC.

In [24], the authors proposed a new TCP variant called TCP-WELCOME that uses Loss Differentiation Algorithm (LDA) and Loss Recovery Algorithm (LRA) techniques to improve the performance over other TCP variants. They used a new methodology for finding RTO value based on old and new RTT values. In [25–27], the authors analyzed the popular asymmetric cryptographic algorithm RSA with Carmichael function, and identified that Carmichael function produces smaller private key than Euler totient function, which reduces the decryption time of the receiver to decrypt the cipher text. In [28,29], the authors proposed fine-tuned values for adaptive TTL to improve performance under rapid changes. Results show that this modified adaptive TTL with dynamic bounds gave good performance than the original.

3. Proposed System

3.1. Enhanced RTT & RTO Calculations in TFRC

TFRC sending rate is primarily a function of estimated average round trip time and loss event rate. TFRC sender has to estimate Retransmission Timeout (RTO) based on round trip time values. However, it does not have fine grained congestion avoidance, because round trip time rapidly changes over the network. Moreover, it increases its sending rate based on packet loss history information,

but it lacks congestion avoidance feature. The RTT measure should be adaptable according to the changes in traffic. If RTO is too small, this results in unnecessary retransmission and leads to wastage of network bandwidth. If RTO is large, this results in high delay and it leads to degradation of network performance. TFRC uses the same kind of approach to find RTO for all packet losses. This same method to find RTO's are not amenable for all situations. We proposed an adaptable RTT and RTO values according to the situation. Congestion or link errors are primary causes of packet losses in the network. TFRC has to differentiate packet loss due to either congestion or link error so that packet losses can be preventable in future. To prevent or control packet loss in the network, RTT and/or RTO values should adjust dynamically. If RTT values are monotonically increasing, then there will be more chances of facing congestion in the network. To avoid packet losses in the network due to congestion, the sender has to maintain k recent RTT samples and follow condition as follows.

Pseudo code for Adaptive RTT & RTO Calculations

$$
\begin{aligned}
&If\left(\left(RTT_{(i+1)} - RTT_{(i)}\right) > Lt\right)\\
&\quad begin\\
&\qquad RTT_{new} = Q * RTT_{old} + (1-Q) * R\_Sample + (RTT_{max} - RTT_{min});?\\
&\qquad RTO = 4 * RTT_{new};\\
&\quad end\\
&else\\
&begin\\
&\qquad RTT_{new} = Q * RTT_{old} + (1-Q) * R\_Sample;\\
&\qquad RTO = 4 * RTT_{new};\\
&\quad end\\
&\\
&Variables :?\\
&Lt : Limited\ threshold,\ RTT_{max} : Maximum\ RTT,\\
&RTT_{min:} Minimum\ RTT;\ RTT_{new} = New\ value\ of\ RTT;\\
&RTT_{old} = Previous\ value\ of\ RTT;
\end{aligned}
\tag{3}
$$

To avoid packet losses due to link error, the RTT and RTO values are updated [24] as follows.

$$
\begin{aligned}
&RTT_{new} = Q * RTT_{old} + (1-Q) * R\_Sample + (RTT_{\max} - RTT_{\min});\\
&RTT_{new} = (RTT_{new}/RTT_{old}) * RTT_{old}
\end{aligned}
\tag{4}
$$

Variables:

RTO_{new} = *New RTO value;* RTO_{old} = *Previous RTO value.*

3.2. Enhanced Average Loss Interval Methods for TFRC

3.2.1. ALI with Confidence Interval

One of the primary functionalities of the receiver is to find loss event rate. Loss event rate consists of several packet losses within a round trip time rather than packet loss rate. For this, the receiver uses Average Loss Interval (ALI) method to find loss event rate. We proposed an enhanced average loss interval method to calculate loss event rate. Advantages of Enhanced ALI method are that it can calculate packet loss more accurately and it reduces the burden on the receiver. The basic ALI method calculates packet loss event rate as follows. Let $\{S_i\}_{i=1,\,2,\,3\,\ldots}$ be the sequence of loss intervals observed by the receiver. S_k is the kth loss event rate with number of events represented by n, the loss event rate can be calculated as $1/\hat{S_{1,n}}$, Where $\hat{S_{1,n}}$ can be calculated as weighted average of the last n intervals as follows.

$$
\hat{S_{1,n}} = \frac{\sum_{i=1}^{n} W_i S_i}{\sum_{i=1}^{n} W_i} \tag{5}
$$

eW_{i} is the weight for each loss history.

$$W_{\mathrm{i}} = f(x) = \begin{cases} 1,\ 1 \le i \le n/2 \\ 1 - \frac{i-n/2}{n/2+1},\ n/2 \le i \le n \end{cases}$$

Generate a sequence $\{\widetilde{S}_i\}_{\mathrm{i}=1,\,2,\,3\ldots}$ as the most recent k sequences of loss intervals derived from set $\{S_i\}_{i=1,\,2,\,3\,\ldots}$ and are updated on the basis of most recent sample. The minimum and maximum loss intervals are obtained using the strategy called confidence interval.

This confidence interval will forecast the future loss interval and will fall under x% confidence interval. In order to find more accurate loss intervals, we replaced the minimum and maximum of most recent k sequences of loss intervals with extreme points of the confidence interval. The lower end of confidence interval can be considered as lower bound and upper end of confidence interval can be considered as upper bound for the most recent k samples $\{S_i^{\sim}\}_{\mathrm{i}=1,\,2,\,3\,\ldots}$

Mathematically Confidence interval can be represented as $(\overline{X} - \mathrm{t}\,\frac{\sigma}{\sqrt{n}}, \overline{X} + \mathrm{t}\,\frac{\sigma}{\sqrt{n}})$. Let $\widetilde{s}_{\mathrm{i(min)}}$ is the lower bound of the sequence $\{S_i^{\sim}\}_{\mathrm{i}=1,\,2,\,3\,\ldots}$ and $\widetilde{s}_{\mathrm{j(max)}}$ is the upper bound of the sequence $\{S_i^{\sim}\}_{\mathrm{i}=1,\,2,\,3\,\ldots}$

$$\left.\begin{array}{rcl} \widetilde{s}_{\mathrm{i(min)}} = & \overline{X} - \mathrm{t}\frac{\sigma}{\sqrt{n}} & \\ \widetilde{s}_{\mathrm{j(max)}} = & \overline{X} + \mathrm{t}\frac{\sigma}{\sqrt{n}} & \end{array}\right\} n - k + 1\langle i.j,\rangle n \tag{6}$$

$\overline{X}$ is the mean of most recent k sequences from the set $\{S_i^{\sim}\}_{\mathrm{i}=1,\,2,\,3\,\ldots}$.
t is the desired confidence coefficient value (1.96 is the default value for 95% confidence interval-t-distribution table), n is the sample size.

Replace the minimum and maximum values in k most recent sample sequence $\{S_i^{\sim}\}_{\mathrm{i}=1,\,2,\,3\,\ldots}$ with $\widetilde{s}_{\mathrm{i(min)}}$ and $\widetilde{s}_{\mathrm{j(max)}}$ respectively. The new average loss interval $\widetilde{S}_{(1,n)}$ can be estimated from $\{S_i^{\sim}\}$. Using the basic ALI method the weighted average of the last n intervals $\widetilde{S}_{(1,n)}$ and loss event rate can be calculated as $1/\widetilde{S}_{(1,n)}$.

$$\widetilde{S}_{(1,n)} = \frac{\sum_{i=0}^{k-1} \mathrm{W}_i \widetilde{S}_{n-i}}{\sum_{i=0}^{k-1} W_i} \tag{7}$$

The corresponding weights for $\widetilde{s}_{i\mathrm{(min)}}$ and $\widetilde{s}_{j\mathrm{(max)}}$ are weights of min and max of k most recent sequences of $\{S_i^{\sim}\}_{\mathrm{i}=1,\,2,\,3\ldots}$ By replacing the min and max intervals from the sequence $\{S_i^{\sim}\}_{\mathrm{i}=1,\,2,\,3}$ with end points of confidence interval in the sequence $\{S_i^{\sim}\}_{\mathrm{i}=1,\,2,\,3\,\ldots}$, the ALI method can produce more accurate and stable estimation of loss event rate than earlier even in network fluctuations. Alternatively, we can include $\widetilde{s}_{i\mathrm{(min)}}$ and $\widetilde{s}_{j\mathrm{(max)}}$ to the k most recent sequence $\{S_i^{\sim}\}_{i=1,\,2,\,3\,\ldots}$ to estimate loss event rate with high probability. Using the basic ALI method

$$\widetilde{S}_{(1,n)} = \frac{\sum_{i=0}^{k+1} \mathrm{W}_i \widetilde{S}_{n-i}}{\sum_{i=0}^{k+1} W_i} \tag{8}$$

3.2.2. ALI with Mean

Let $\{Si\}_{i=1,\,2,\,3\,\ldots}$ be the sequence of loss intervals observed by the receiver. Generate a sequence $\{\widetilde{S}_i\}_{i=1,\,2,\,3\ldots}$ as the most recent k sequences of loss intervals derived from set $\{S_i\}_{i=1,\,2,\,3\,\ldots}$ and are updated on the basis of most recent sample. Let us denote $\mathrm{S_{min}}$ and $\mathrm{S_{max}}$ are the minimum and maximum values of the sequence $\{S_i^{\sim}\}_{i=1,\,2,\,3\,\ldots}$ Find out the mean value of all loss intervals from the sequence $\{S_i^{\sim}\}_{i=1,\,2,\,3\,\ldots}$ and is denoted by $\mathrm{S_{mean}}$. Replace the minimum $\mathrm{S_{min}}$ and maximum $\mathrm{S_{max}}$ values from the sequence $\{S_i^{\sim}\}_{i=1,\,2,\,3\,\ldots}$ with $\mathrm{S_{mean}}$ individually. Now the sequence $\{\widetilde{S}_i\}_{i=1,\,2,\,3\ldots}$ contains k most recent samples with two new values. These result to estimate average loss intervals more accurately with fewer samples. The corresponding weights for $\mathrm{S_{mean}}$ are weights of minimum $\mathrm{S_{min}}$ and maximum $\mathrm{S_{max}}$ of k most recent sequences of $\{S_i^{\sim}\}_{i=1,\,2,\,3\,\ldots}$

Using the basic ALI method, the weighted average of the last n intervals $\widetilde{S}_{(1,n)}$ and loss event rate can be calculated as $1/\widetilde{S}_{(1,n)}$. Mathematically represented as

S_{min} = minimum $\{S_i^{\sim}\}_{i=1,\,2,\,3\,\ldots}$ S_{max} = maximum $\{S_i^{\sim}\}_{i=1,\,2,\,3\,\ldots}$

$$S_{mean} = \frac{\sum_{i=1}^{k} S_i}{k} S_i \,\epsilon\, \{S_i^{\sim}\}_{i=1,\,2,\,3}\ \widetilde{S}_{(1,n)} = \frac{\sum_{i=0}^{k-1} W_i \widetilde{S}_{n-i}}{\sum_{i=0}^{k-1} W_i} \tag{9}$$

Alternatively, we can include S_{mean} into the sequence $\{S_i^{\sim}\}_{i=1,\,2,\,3\,\ldots}$ along with S_{min} and S_{max}, then weighted average of the last n intervals $\widetilde{S}_{(1,n)}$ can be calculated as

$$\widetilde{S}_{(1,n)} = \frac{\sum_{i=0}^{k} W_i \widetilde{S}_{n-i}}{\sum_{i=0}^{k} W_i} \tag{10}$$

3.3. Reduce Packet Loss with TTL

A few packets are dropped due to loops in the networks. Sometimes packets have to travel a greater number of routers than usual. This may result in packets being not delivered to the destination within a reasonable length of time. There are some mechanisms to avoid congestion in the network and to redirect the packets to nearby sub routers. When a packet travels a greater number of routers unexpectedly, its TTL value goes to zero before reaching the packet to the destination. This motivates the proposal of an adaptive TTL so that even when the packet travels more routers than usual, it should not be discarded by any router. This avoids unnecessary retransmissions in the network and it leads to reduced packet losses. An adaptive TTL can be calculated as follows.

$$\left.\begin{array}{c} TTL_{new} = \mathrm{Max}(TTL_{min},\ \mathrm{Min}(TTL_{rt},\ TTL_{max})) \\ TTL_{rt} = \alpha * TTL_{est(i)} + (1-\alpha) TTL_{est(i-1)} + \left(TTL_{est(i)} - TTL_{est(i-1)}\right) \end{array}\right\} 0.5 \le \alpha < 1 \tag{11}$$

Variables:

TTL_{min} = static specified minimum value; TTL_{max} = static specified maximum value
TTL_{rt} = Recent reflect changes;
$TTL_{est(i)}$, $TTL_{est(i-1)}$ are the two most recent observations.

Sometimes increasing the TTL value may lead to security problems in the network because with increased TTL packet can traverse a longer route. There might be chances that with the increased TTL, the packet can enter into an unauthorized private network. Security mechanisms have to be provided to avoid security problems for an increased TTL packet. For this, popular asymmetric cryptographic algorithm, RSA with Carmichael function is applied to each increased TTL packet or any packet that needs security over the network. The advantage of this encryption scheme is that it produces small private key value compared with usual Euler totient function in RSA. The original recipient can decrypt the cipher text with less time than usual. This helps to prevent security threats for any increased TTL packet.

4. Simulation Environment

Network simulator NS2.35 is one of the popular simulation tools which are used to understand and predict the protocols behavior in networks. In order to simulate congestion control protocols, dumbbell topology is a good model to reflect real-world environment. For this, we have chosen dumbbell topology with multiple bottleneck links consisting of ten nodes with different bandwidths (5 Mb, 4 Mb, 10 Mb, 2 Mb) incorporating delay of 10 ms as shown in Figure 2. Packet size of TFRC is 1000 bytes and total simulation time is 175 s.

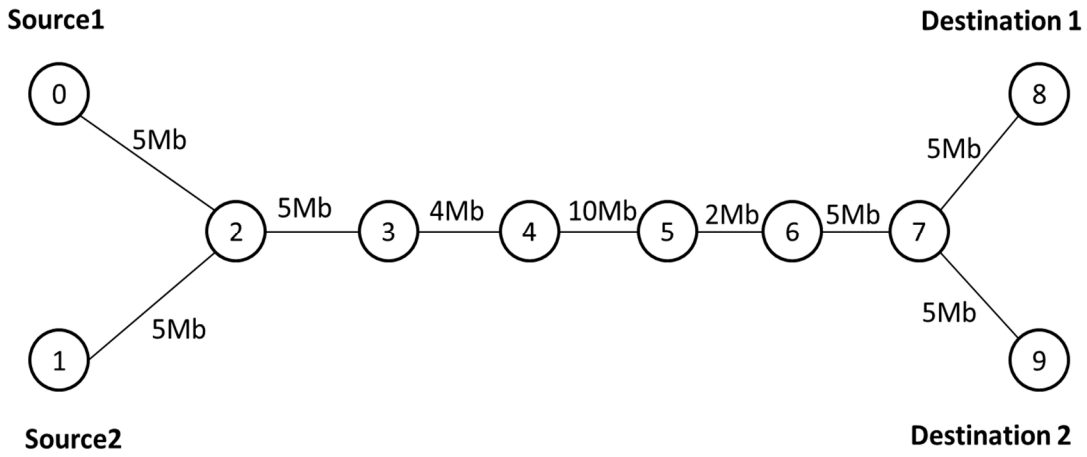

Figure 2. Dumbbell topology.

5. Results and Analysis

- Throughput: Throughput measures how fast data can be sent across the network.
- Packet Loss Rate: It can be measured as the percentage of number of packets dropped or lost by the number of packets sent over the link.
- End-to-End delay: It represents the responsiveness of the network. It can be measured as average over all surviving data packets from the sources to destinations.
- Packet Delivery Ratio: It is the ratio between the data packets delivered by the destination to number of packets generated by the sources.
- Jitter: It represents the variation of the packet arrival time. It can be calculated as the difference between the delay of the current packet and previous packet. The value of jitter should be less for streaming applications like audio or video applications.

We have applied the above key strategies to TFRC and called it enhanced TFRC and we compared the performance of enhanced TFRC and classical TFRC in terms of throughput, packet loss rate, jitter and average end to end delay.

Figure 3 depicts the throughput efficiency of enhanced TFRC and classical TFRC. With increasing data rate, enhanced TFRC produces better throughput than classical TFRC. Enhanced TFRC possesses best traffic engineering principles with effective packet loss method. When comparing the performance of enhanced TFRC and classical TFRC in terms of throughput, enhanced TFRC results in 22% more throughput than classical TFRC. Figure 4 illustrates the comparison of packet loss ratio between enhanced TFRC and classical TFRC. With increasing data rate, enhanced TFRC results in low packet loss ratio than classical TFRC due to the proper distinction of packet losses with enhanced round trip time and retransmission timeouts. When comparing the performance of enhanced TFRC and classical TFRC in terms of packet loss ratio, enhanced TFRC results in 20% less packet loss ratio than classical TFRC. Figure 5 shows a comparison of enhanced TFRC and classical TFRC in terms of End-to-End delay. Enhanced TFRC results in less average end-to-end delay over classical TFRC by 20% due to enhanced feature of retransmission of timeout and enhanced TTL. Figure 6 shows the result of packet delivery ratio. Enhanced TFRC performs better than existing TFRC by 22%. Figure 7 shows a comparison of jitter between enhanced TFRC and classical TFRC. Enhanced TFRC produces less packet variation compared with classical TFRC by 18%.

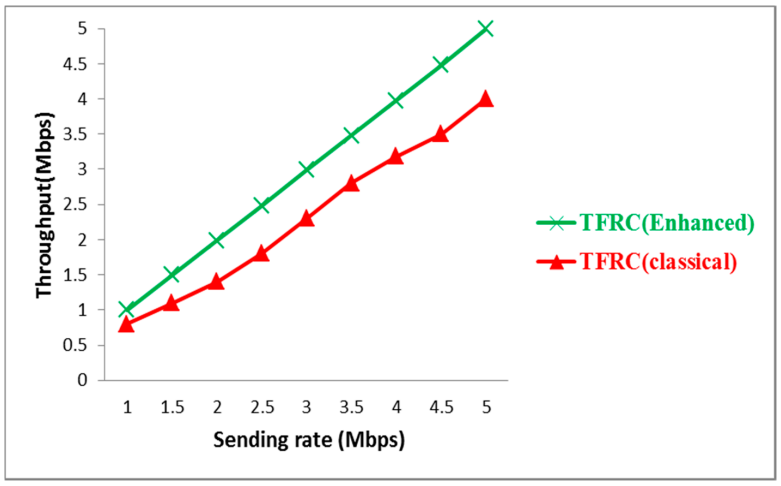

Figure 3. Throughput comparison of classical TFRC and enhanced TFRC.

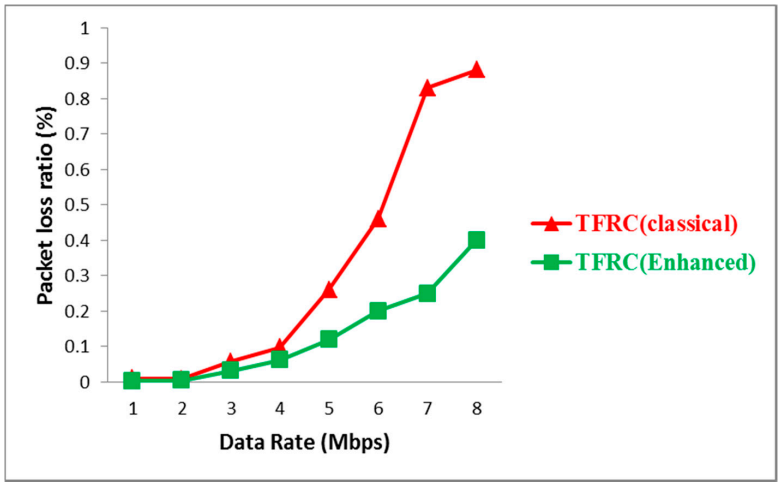

Figure 4. Comparison of packet loss ratio between classical TFRC and Enhanced TFRC.

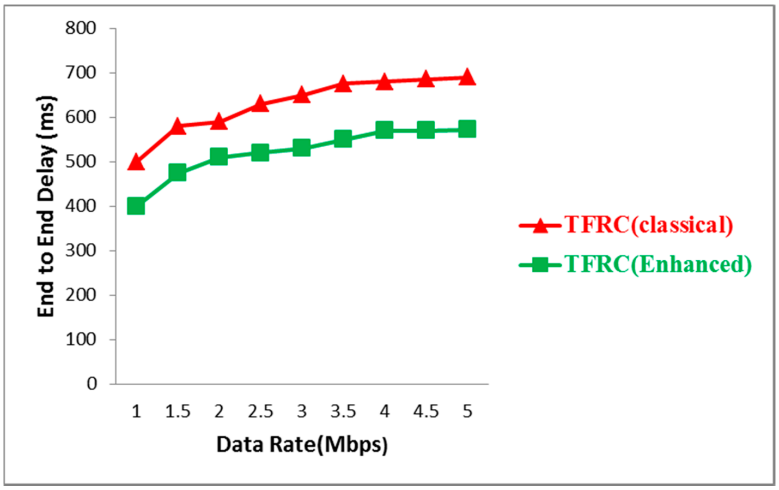

Figure 5. Comparison of End-to-End delay between classical TFRC & Enhanced TFRC.

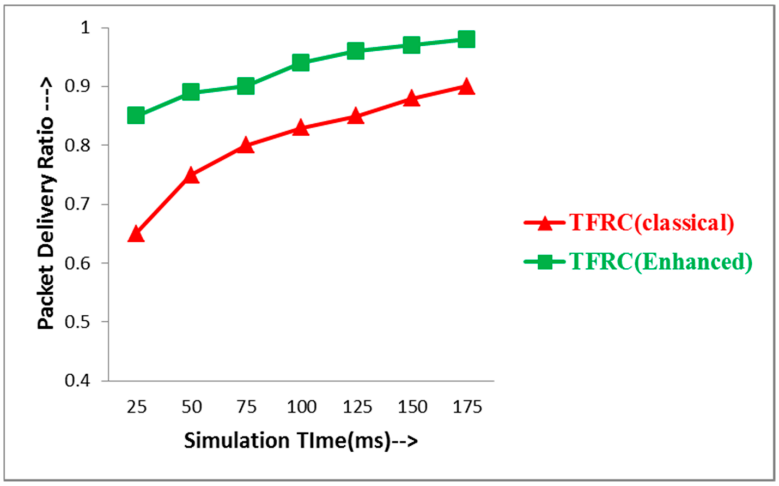

Figure 6. Comparison of packet delivery ratio between classical and enhanced TFRC.

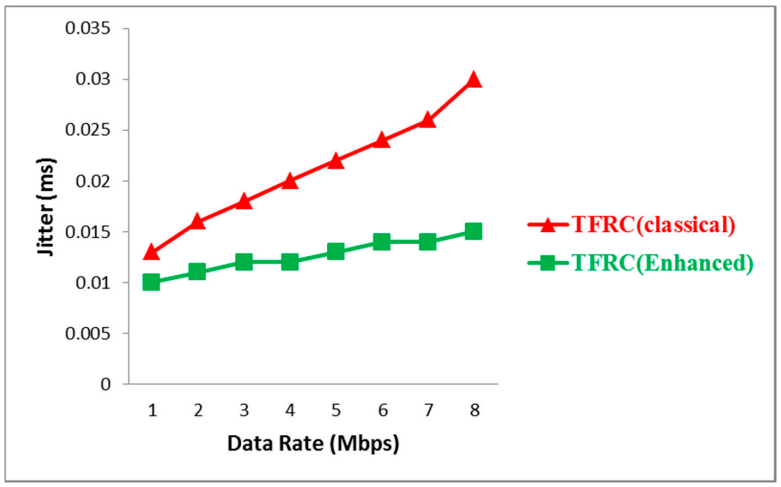

Figure 7. Comparison of jitter between classical TFRC & enhanced TFRC.

6. Conclusions

Congestion control mechanisms play a prominent role for transmitting real time multimedia applications. TFRC aims to transfer multimedia applications by maintaining a smooth sending rate over the network. However TFRC is unable to produce better results in all possible environments. It needs some improvements in order to be used for multimedia traffic. This paper presents efficient traffic engineering strategies and is applied to TFRC to alleviate the problems of packet losses due to congestion and link errors. An enhanced TFRC includes adaptive RTT and RTO calculations, efficient average loss interval method for finding packet loss rate and adaptive TTL feature for reducing packet loss. Moreover, enhanced TFRC helps to improve the performance by reducing unnecessary retransmissions with the inclusion of enhanced RTT, RTO and TTL features. Simulation results have shown that enhanced TFRC produces increased throughput and less packet loss even with high network traffic, which is useful especially for transferring VOIP and online streaming video applications. As future work, we plan to measure TFRC throughput in adversities like measuring round trip time during dynamically changing bandwidth. Moreover, we plan to differentiate sensitive information by TFRC and establish efficient methods for sending variable lengths of data.

Author Contributions: Nalavala Ramanjaneya Reddy created the main ideas and wrote the paper; Pakanati Chenna Reddy and Mokkala Padmavathamma critically reviewed and revised the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Jacobson, V. Congestion Avoidance and Control. In Proceedings of the ACM Sigcomm, Scanford, CA, USA, 26–30 August 1988; pp. 314–329.
2. Floyd, S.; Handley, M.; Padhye, J.; Widmer, J. Equation-based congestion control for unicast applications. In Proceedings of the ACM SIGCOMM 2000, Stockholm, Sweden, 28 August–1 September 2000; pp. 43–56.
3. Thompson, K.; Miller, G.J.; Wilder, R. Wide-area Internet traffic patterns and characteristics. *IEEE Netw.* **1997**, *11*, 10–23. [CrossRef]
4. Floyd, S.; Henderson, T.; Gurtov, A. The New Reno Modification to TCP's Fast Recovery Algorithm. IETF: Fremont, CA, USA, 2004.
5. Floyd, J.P.S.; Handley, M.; Padhe, J.; Widmer, J. *Protocol Specification: TCP Friendly Rate Control (TFRC)*; IETF: Fremont, CA, USA, 2008.
6. Chodorek, A.; Chodorek, R.R. Streaming Video over TFRC with Linear Throughput Equation. *Adv. Electron. Telecommun.* **2010**, *1*, 26–29.
7. Vasallo, P.R. *Variable Packet Size Equation-Based Congestion Control*; ICSI Technical Report; ICSI: Berkeley, CA, USA, 2000; pp. 1–11.
8. Zhou, B.; Fu, C.P.; Li, V.O.K. TFRCVeno: An enhancement of TCP friendly rate control protocol over wired/wireless networks. In Proceedings of the IEEE International Conference on Network Protocols, Beijing, China, 16–19 October 2007; pp. 216–225.
9. Wei, S.; Wen, T.; Guo, Q. Improving the Start-up Performance of the TFRC Protocol. *Comput. J.* **2012**, *56*, 1269–1278.
10. Arjuna, S.; Fairhurst, G. Use of quick start for improving the performance of TFRC-SP over satellite networks. In Proceedings of the International Workshop on Satellite and Space Communications, Madrid, Spain, 14–15 September 2006; pp. 1–5.
11. Li, M.; Lee, C. Emmanuel Agu, Mark Claypool, and Robert Kinicki, Performance enhancement of TFRC in wireless ad hoc networks. In Proceedings of the 10th International Conference on Distributed Multimedia Systems (DMS), San Francisco, CA, USA, 8–10 September 2004; pp. 1–6.
12. Cho, S.; Woo, H.; Lee, J. ATFRC: Adaptive TCP Friendly Rate Control Protocol. In Proceedings of the ICOIN, Cheju Island, South Korea, 12–14 February 2003; Springer: Berlin/Heidelberg, Germany, 2003; pp. 171–180.
13. Nahm, K.; Kuo, C.-J. Design and performance evaluation of TCP-friendly thin-layerd video multicast scheme. In Proceedings of the IEEE ICME 2004, Taipei, Taiwan, 27–30 June 2004; pp. 487–490.
14. Nahm, K.; Kuo, C.-J. Low-variance TCP-friendly throughput estimation for congestion control of layered video multicast. In Proceedings of the ISCAS 2004, Vancouver, BC, Canada, 23–26 May 2004; pp. 209–212.
15. Reddy, N.R.; Reddy, P.C.; Padmavathama, M. Performance Enhancement of TCP Friendly Rate Control Protocol over Wired networks. *Int. J. Electr. Comput. Eng.* **2016**, *6*, 2949–2954. [CrossRef]
16. Reddy, N.R.; Reddy, P.C.; Padmavathama, M. An Enhanced Queue Management Scheme for TFRC over Wired Networks. *Int. J. Intell. Eng. Syst.* **2017**, *10*. [CrossRef]
17. Li, D.; Sleurs, K.; Lil, E.V.; de Capelle, A.V. Improving TFRC Performance against Bandwidth Change during Handovers. In Proceedings of the 4th International Conference on Wireless Communications, Networking and Mobile Computing, ICWCNM 2008, Dalian, China, 12–14 October 2008; pp. 1–4.
18. Lee, S.; Chung, K. Enhanced TFRC to improve the quality of multimedia streaming service. In Proceedings of the International Conference on ICT Convergence, Jeju Island, South Korea, 15–17 October 2012; pp. 373–378.
19. Ullah, I.; Shah, Z.; Baig, A. S-TFRC: An Efficient Rate Control Scheme for Multimedia Handovers. *Comput. Sci. Inf. Syst.* **2016**, *13*, 45–69. [CrossRef]
20. Oh, B.H.; Han, J.; Kim, K.; Lee, J. A New Receiver-Based Retransmission Scheme with TFRC. *IEEE Commun. Lett.* **2012**, *16*, 2091–2094. [CrossRef]
21. Lee, S.; Roh, H.; Lee, H.; Chung, K. Enhanced TFRC for high quality video streaming over high bandwidth delay product networks. *J. Commun. Netw.* **2014**, *16*, 344–354. [CrossRef]
22. Song, Z.; Zhang, Y.; Zhou, M. Enhanced TFRC congestion control mechanism based on refined network states division. In Proceedings of the IEEE China Summit and International Conference on Signal and Information Processing, ICSIP, Chengdu, China, 12–15 July 2015; pp. 938–942.

23. Sterca, A.; Hellwagner, H.; Boian, F.; Vancea, A. Media-Friendly and TCP-Friendly Rate Control Protocols for Multimedia Streaming. *IEEE Trans. Circuits Syst. Video Technol.* **2016**, *26*, 1516–1531. [CrossRef]
24. Ghaleb-Seddik, A.; Ghamri-Doudane, Y. Coupling Loss and Delay Differentiation to Enhance TCP Performance within Wireless Multi-hop Ad-hoc Networks. *J. Commun.* **2012**, *7*, 859–872. [CrossRef]
25. Rivest, R.; Shamir, A.; Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **1978**, *21*, 120–126. [CrossRef]
26. Carmichael, D.R. Note on a new number theory function. *Bull. Am. Math. Soc.* **1910**, *16*, 232–238. [CrossRef]
27. Ramanjaneyareddy, N.; Reddy, P.C.; Padmavathamma, M. Study the effect of Carmichael function on RSA. In Proceedings of the SmartCom 2016, Jaipur, India, 6–7 August 2016; Springer: Singapore, 2016; pp. 752–756.
28. Srinivasan, R.; Liang, C.; Ramamritham, K. Maintaining Temporal Coherency of Virtual Data Warehouses. In Proceedings of the IEEE Real-Time Systems Symposium, Madrid, Spain, 4 December 1998; pp. 60–71.
29. Cheng, A.M.K.; Zhang, Z. Improving Web Server Performance with Adaptive Proxy Caching in Soft Real-time Mobile Applications. *J. VLSI Signal Process.* **2007**, *47*, 103–115. [CrossRef]