Original Article

AN IMPROVED CONGESTION CONTROL STRATEGY FOR TFRC OVER HETEROGENEOUS NETWORKS

Rohit Kumar Sharma

Computer Engineering Department,
National Institute of Technology (NIT),
Kurukshetra, Haryana, India
DOI:https://doi.org/ 10.5281/zenodo.17099618

ABSTRACT: For advanced streaming applications over wired-wireless networks TCP-Friendly Rate Control (TFRC) has been widely adopted nowadays to give smooth sending rate and unceasing quality in streaming applications TFRC applies an equation-based rate control scheme. However, TFRC tends to fail in wireless environment if packet lost event was done by poor channel quality but network congestion. Therefore, TFRC not able to provide high quality-of-service for streaming applications over wired-wireless networks. In this paper, we proposed a delay based uni-directional delay jitter based TFRC with end-to-end semantic over wired-wireless networks. This scheme provide smooth sending rate and TCP friendly characteristics like standard TFRC, even it also increase the throughput by estimating the available bandwidth in wired-wireless networks with burst nature of background traffic. Simulation results show performance improvement without intrusiveness issue and even if background traffic is bursty over wired-wireless networks.

Keywords: Bursty network Congestion Control Mechanism, Streaming applications, TCP Friendly Rate Control (TFRC), Uni Directional Delay Jitter

I. Introduction

In the recent years, wireless communication technologies ease the deployment of broadband network access and facilitate Internet access anytime, anywhere. Increase in bandwidth of wireless communication systems, subscribers can provide variety of multimedia services like streaming applications. User Datagram Protocol (UDP) is most common transport protocol for the streaming applications. UDP flows could collapse the networks because UDP transmit data as fast as it can without rate or congestion control mechanism. UDP is also not a TCP-Friendly protocol. Therefore, UDP is not a best suited transport protocol for streaming application with quality-of-service issue. To achieve quality-of-service of streaming applications, many rate control schemes were proposed. A most widely adopted rate control scheme is TCP Friendly Rate Control (TFRC) [5]. TFRC provide network stability, fairness and smooth sending rate. By using an equation-based rate control, TFRC keeps TCP-friendly to avoid collapsing network. Although TFRC able to provide smooth perceptual quality in streaming applications, it was taken as conservative in competing with background TCP traffic to keep TCP-friendly.

Original Article

Without using higher priority in sending streaming applications, the bandwidth requirement of streaming applications is hard to achieve when it competes with TCP traffic. Therefore, streaming applications needs both smooth sending rate and prioritized application QoS. Number of researches were devoted to the issue of performance degradation over wired-wireless heterogeneous network. These researches can be roughly classified into two types of heuristics [1]. To get channel loss information from intermediate node [2] [3] [11]. Intermediate node likes access point can provide required information of channel. The second type of approaches modify the transport protocol to estimate congestion signal by filtering channel loss from packet lost event [4] [8] [10].In wired wireless networks, packet loss events are composed of congestion and channel loss. In order to detect congestion loss accurately, they use delay or round-trip time to distinguish channel loss from observed packet lost events. Both delay and round-trip time carry not only queuing delay of bottleneck but also measurement noise. Therefore, they use kinds of statistic methods to filter out measurement noise at expensive of coarse resolution. In this paper, we proposed a new end-to-end scheme, UDDJ-TFRC, to achieve (1) Prioritized application QoS: For real-time applications, transport protocol should be able to provide higher throughput in competing with background data service without intrusiveness issue. (2)Resistance in channel loss: When streaming applications sent by TFRC, its throughput will be limited by TFRC's congestion control mechanism and channel loss issue. (3) No scalability issue: It is an end-to-end approach without intermediate node support. The rest of paper is organized as follows. In Section 2, we will

Discuss the details of the proposed method. In Section 3, we use NS-2 simulation to show the performance improvement and no intrusiveness of proposed method. Conclusions and future works are discussed in Section 4

II. Proposed Scheme

Unit Directional Delay jitter-TFRC (UDDJ).Rate control mechanism of UDDJ-TFRC is inherited from TFRC to keep the advantage of smooth rate control. The difference between UDDJ-TFRC and TFRC is only at the receiver side. Figure 1 is the overview of UDDJ-TFRC. UDDJ-TFRC will activate each round-trip time or packet loss detected. In the following article, we will introduce relationship between unit directional delay jitter and queuing delay jitter first and later discuss how to predict network congestion from queuing delay jitter in section 3.2. By using congestion prediction system in Figure 1(a), UDDJ-TFFRC can provide prioritized application QoS by utilizing residual bandwidth and avoid collapsing network before bottleneck router dropping incoming packets

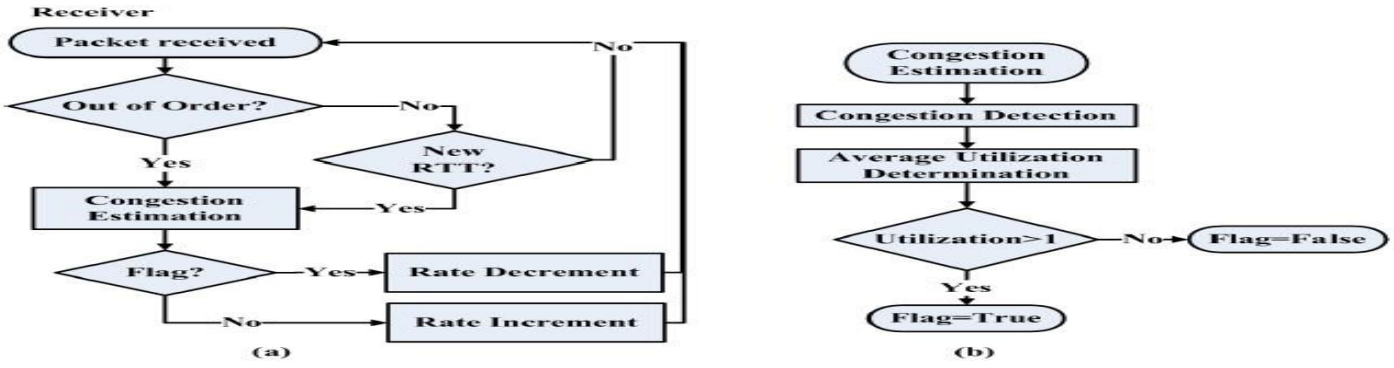


Fig.1 overview of UDDJ (a) Receiver behavior (b) Congestion estimation

Original Article

2.1 Uni Directional Delay Jitter

If the sending rate of UDDJ-TFRC connection is higher than the residual bandwidth, then the outgoing data packets will start to queue in the bottleneck router, resulting in increasing uni directional delay. The uni directional delay jitter, ΔUDD^i, measured as the difference between two uni-directional delays, can be derived as

$$\Delta UDD^i = (\tau^i - \tau^{i-1}) + (\omega^i - \omega^{i-1}) + (\beta^i - \beta^{i-1}) \quad (1)$$

Which can be further simplified as

$$\Delta UDD^i = \beta^i - \beta^{i-1} \quad (2)$$

Where denotes transmission delay, ω denotes service delay , based on these assumptions that the difference between transmission delays can be remove if the packets are transferred over the same path and that the difference between delays can be removed if the packet size is the same. In multi-hop topology with a single bottleneck environment, let the propagation path be composed of P links, and let each hop, hop_p ($1 \le p \le P$), consist of a FIFO queue and a store and forward router. The uni-directional delay jitter of the multi-hop path can be obtained by summing the uni directional delay jitters (Eq. (2)) of all the hops for the i-th consecutive data packets, which is simplified due to domination of the bottleneck as

$$\Delta UDD^i = \beta_b^i - \beta_b^{i-1} \quad (3)$$

Where b ($\in$ [1 P]) denotes the location of bottleneck and $\beta_p^i - \beta_p^{i-1} = 0$ holds for $p \neq b$. In Eq. (3), it is clear that uni directional delay jitter carry out the information of queuing of data packets, which can be used to predict the status of bottleneck router.

2.2 Congestion Prevention

In this section, we discuss how to use uni directional delay jitter to know congestion before congestion packet loss detected. Initially, let outgoing data packets be composed of n packet pairs with same packet size and sending rate. Let PP^i ($1 \le i \le n$), which is composed of consecutive two data packets P^{i-1} and P^i, denote the i-th packet pair, whose inter-departure time is denoted by Δ_{id}, so-called input gap. Let ub denote the utilization of bottleneck router during the inter-arrival time of PP^i. If $ub \ge 1$, then the bottleneck router does not finish the following three tasks before the arrival of P^i: processing the queued traffic (let θ^{i-1} denote the sum of the queued traffic and undelivered packets when P^{i-1} arrives at the router), processing P^{i-1} (let s be the constant packet size), and processing the cross traffic inserted within this packet pair PP^i (let C^i denote the amount of cross traffic arriving at the bottleneck during the inter-arrival time of PP^i). These tasks can be related to each other as follows:

$$\left(\frac{\theta^{i-1} + s + C^i}{B}\right) > \Delta id, if\ ub^i \ge 1 \quad (4)$$

Where B denotes the bandwidth of bottleneck. We use, so-called output gap, to denote inter-arrival time of a packet pair PP^i at the receiver side, which is equal to. In addition, let $\frac{\theta^{i-1}}{B}$ be denoted as β^{i-1}, representing the queuing delay that accumulated before packet P^{i-1}. Thus, Eq. (4) can be rewritten as

$$\Delta_{og}^i > (\Delta id - \beta^{i-1}), if\ ub^i \ge 1 \quad (5)$$

Eq. (5) can be further rewritten by substituting the uni directional delay jitter for the difference between the output and input gaps as follows:

$$\Delta SPD^I + \beta^{I-1} > 0,\ if\ ub^I \ge 1, \quad (6)$$

Where $\Delta UDD^i = \Delta_{og}^i - \Delta_{id}$

Original Article

Eq. (6) shows what can be derived if bottleneck is congested during the
Inter-arrival time of data packets. Delay jitter and the accumulated queuing delay in order to determine ub^i. In addition, Eq. (3) also revealed that uni directional delay jitter closely depends on the accumulated queuing delay, which implies that an correct queuing delay propagation mechanism is indispensable. We will discuss this issue later. Thus, the condition that can be used to determine ub^i is defined as

$$ub^i \geq 1; \quad \text{if} \quad \Delta UDD^i + \beta^{i-1} > 0; \tag{7}$$
$$ub^i = 0; \quad \text{Otherwise.}$$

Here, we assume bottleneck router is idle or full-utilized during the inter-arrival time of data packets. However, if noise existed in Δ_{log}^i, ub^i may be false-estimated. Once ubi is determined based on Eq. (9), the accumulated queuing delay can be β^i determined by following rule:

$$\beta^i = \begin{cases} \beta^{i-1} + \Delta UDD^i, if\ ub^i = 1; \\ max(0, \Delta UDD^i), otherwise \end{cases} \tag{8}$$

As a result, if the procedures shown in Eqs. (7) And (8) are iteratively performed, then we can get an average utilization of bottleneck. Obviously, the iterative procedure can't process if the information of queuing delay of first data packet, so-called initial queuing delay, is unknown. Since the accumulated queuing delay and uni directional delay jitter play key roles in determining the type of queuing region, they are also exploited to determine the initial queuing delay. We only require to trace the output gaps of a packet train once to determine the initial queuing delay. We will investigate this issue based on two cases.

In the first case, the first uni directional delay jitter is negative, i.e., $\Delta UDD^1 < 0$. It means that initially the first output gap is reduced to absorb the initial queuing delay. The size of the first output gap depends on two points: the queuing delay and cross traffic. If there is no cross traffic or the amount of inserted cross traffic is not large enough to expand the first output gap, then $|\Delta UDD^1|$ is equal to the initial queuing delay β^0. On the other hand, if $\beta^0 > |\Delta UDD^1|$, then subsequent output probing gaps will be gradually reduced so as to absorb β^{k-1}. Propagation of the queuing delay will continue, and a decreasing sequence of queuing delays, $\beta 0 > \beta^1 > \cdots >$, will be generated until $< \beta^k$, which is caused by the inserted cross traffic, is satisfied. It means that the initial queuing delay will be completely exhausted. Therefore, the initial queuing delay can be calculated and found to be the absolute value of β^{k-1}, $|\beta^{k-1}|$.

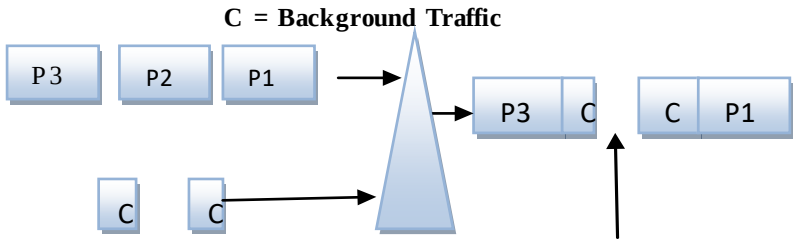
In the second case, the first uni-directional delay jitter is larger than zero, i.e., $\Delta UDD^1 > 0$. In this situation, we cannot be sure whether the initial queuing delay will be completely absorbed in the first output gap or if it does not exist initially. However, we can infer the queuing delay β^1 of the second packet based on Eq. (8) and use it as though it were the initial queuing delay, even though we will lose the aggregated traffic captured in the first output gap. This phenomenon might affect the average utilization determination slightly. However, its shortterm impact on the accuracy of available bandwidth estimation can be inconsiderable when compared with the long-term impact of noise. After the instant utilization and queuing delays in the data packet stream have been determined, average utilization can be computed to determine the cause of loss event.

2.3 Congestion Detection

Before getting into detail how to detect congestion under channel packet loss, we use Fig 2 to look what happen if router is congested but packet dropped by wireless link. In Fig 2, router was congested during the inter-arrival time of P1 and P3. If P2 received at receiver side successfully, the

Original Article

P = UDDJ-TFRC Packets



WIRELESS LOSS

Fig.2 Data packet P2 was lost caused by wireless link and router is congested by UDDJ-TFRC and aggregate TFRC

DATA PACKET

BACKGROUND TRAFFIC

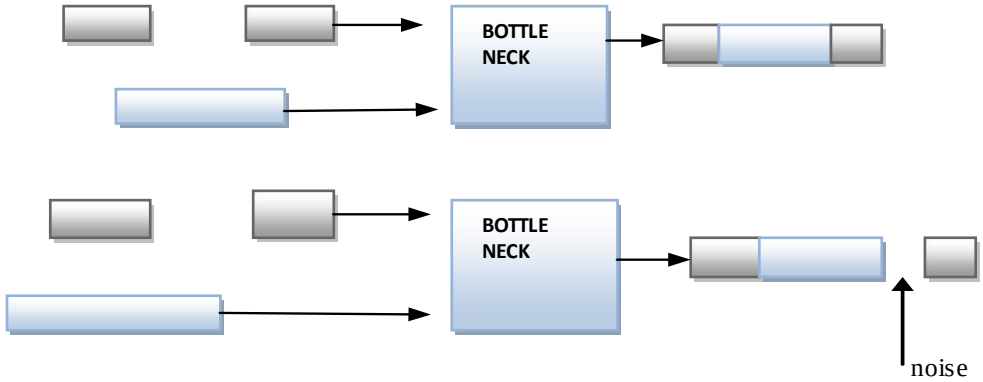


Fig.3 The probing of appearing noise which are gaps that are not filled with any traffic data

Relationship among uni directional delay jitter, queuing delay jitter will same as Eqs. (7) and (8). Therefore, and $\Delta SPD^1 + \beta^0 > 0$ will be estimated and by Eq. (10) In Fig 2, we $\Delta UDD^2 + \beta^1 > 0$ $\beta^1 = \beta^0 + \Delta UDD^1$

Use Δ to denote uni directional delay jitter of P1 and P3 if P2 was lost caused by poor channel quality. If router was congested as the example shown in Fig 2, Δ will be equal to the sum of two uni directional delay jitters, $UDD^{1'} + \beta^0 > 0$ ΔUDD^1 And ΔUDD^2. On the contrary, if ΔUDD$+ \beta^0 > 0$ was estimated but router was not congested caused by probe noise, which is shown in Figure 3, UDDJ-TFRC will false-estimate congestion status during inter-departure time between P1 and P3. To solve this problem, we need to further discuss probe noise and how to improve accuracy of UDDJ-TFRC's congestion detection under wireless environment. When bursty cross traffic is encountered, the inter-arrival time of data packets at receiver, even those carrying useful information for measuring the cross traffic rate, are contaminated with noises, which are defined as gaps that are not occupied by traffic. An example of noise is illustrated in Figure 3, where the aggregated traffic's rate for the two scenarios is larger than the bottleneck bandwidth. Consequently, bottleneck router should be congested during the inter-arrival time of a packet pair. However, a bottleneck in the second scenario was not congested due to the appearance of probing noise. Liu et al. [7] pointed out that noise is caused by the superposition of incoming packet and background traffic even if the aggregated rate is larger than the Bottleneck bandwidth. Let the noise item be denoted by ε^i .The average utilization of two consecutive data packet can be established [7]:

Original Article

both sides of Eq. (9). The average utilization, , can be get by summing $\frac{\Delta^i_{bg}}{\Delta_{id}}$ in which $ub^i \geq 1$. we can derive

$$ub^i = \frac{\Delta^i_{bg}}{\Delta_{id}} - \frac{\varepsilon^i}{\Delta_{id}} \quad (9)$$

$$\mu \geq \sum_{i=1}^{n}\left(\frac{\Delta^i_{bg}}{\Delta_{id}}\right) \geq \frac{\sum_{i=1}^{n}\Delta^i_{og|ub^i\geq 1}}{n\times\Delta_{id}} \quad (10)$$

In order to

$$\frac{\sum_{i=1}^{n}\Delta^i_{og|ub^i\geq 1}}{n\times\Delta_{id}}$$

probing noise exist. Thus, congestion can be detected by following rules :

Congestion Flag = True, if $\frac{\sum_{i=1}^{n}\Delta^i_{og|ub^i\geq 1}}{n\times\Delta_{id}}$ (11)

estimate the average utilization of bottleneck router when n data packets arrived at receiver, by summing

μ

As we know, If ≥ 1, then average bottleneck utilization, eu, will be larger than 1 even if Congestion Flag = False, otherwise:

By using above iterative procedure, UDDJ-TFRC can detect congestion by estimating one-way delay jitter from received packets. This congestion flag will be updated every round-trip time or if packet loss was detected.

III. Simulation

For the improvement of modified TFRC by the proposed method we discuss simulation in this section. At very first we modeled the throughput of TFRC in wireless Environment. Throughput of TFRC is determined by the rate model in [5] and can be represented as $\frac{ks}{rtt\times\sqrt{p}}$, where k is constant factor and s is the packet size. rtt is

Round-trip-time and p denotes probability of end-to-end loss. When TFRC operated in wireless environment, end-to-end loss is composed by wireless and congestion loss and its throughput will be degraded as:

$$throughput = \frac{ks}{rtt\times\sqrt{p+(1-p_w)\times p_c}} \quad (12)$$

Where pw and p_c denote wireless and congestion loss. It can be find out the upper bound of TFRC's throughput exists while $p_w = p_c = 0$, and lower bound of TFRC's throughput in Eq. (12) under wireless environment is known by p_w if no congestion loss, $p_c = 0$. However, the channel loss only can be calculated under intermediate node support. Hence, the improvement of end-to-end approaches is based on detecting congestion loss, p_c. Here we use topology shown in Figure 4.(a) to calculate the comparison in the improvement of proposed method, TFRC and Wireless-TFRC, which shows that it can distinguish all channel loss from estimated loss events. We did NS-2 simulations [9] to calculate the performance of UDDJ-TFRC. The background traffic are made up of three CBR flows with 6Mbps aggregated throughput and the wireless bandwidth is set be 10Mbps. As a result, the optimal throughput is 4Mbps, which is also the residual bandwidth. The random loss rate in wireless link varies from 0 to 0:08 in increments of 0:005. Drop Tail queue policy is taken for each router and bottleneck is the wireless link. We compare the throughput behavior of WirelessTFRC, TFRC, TCP-RENO and UDDJ-TFRC. The results of

Original Article

average throughput obtained using WirelessTFRC, TFRC, TCP-RENO and UDDJ-TFRC are plotted in Figure 4. The background traffic is CBR traffic with constant throughput, therefore, the congestion event is only because of transport protocols. When random loss ratio is 0%, the loss event is only made of congestion loss, which is raised by transport protocol. Under this condition, throughput of Wireless-TFRC and TFRC are the similar. As the increasing of random loss ratio, throughput is decreased as the increasing p_w. However, by using uni-directional delay jitter congestion detection, UDDJ-TFRC can approximate the optimal throughput to provide better application QoS. The optimal throughput in Figure 4 denotes the maximum throughput without arising congestion loss, p_c, in Eq. (12). Besides, TFRC performs better than TCP in this scenario because background traffic is CBR traffic only such that calculated loss event of TCP and TFRC are different. The results in Figure 5 also shows that UDDJ-TFRC can detect congestion event correctly even if wireless loss exists. Following simulations, calculated the performance of UDDJ-TFRC by using more complicated topologies and bursty background traffic. For all simulations, the wireless bandwidth was set as 10Mbps. Drop Tail queue policy was taken for each router. In order to calculate performance of UDDJ-TFRC under channel loss, we observe its throughput behavior in two multi-hop topologies includes path-persistent and one-hop persistent. We compared the average throughput of TCP-Westwood, JTCP and UDDJ-TFRC for 2000 seconds in NS-2.

3.1 Path Persistent Topology

We specifies a multi-hop with single bottleneck environment, where we have sender and receiver, and 3 types of traffic at senders and 1 at receiver in the cross traffic. We assume no random packet loss in wired links. The cross traffics are composed of three CBR traffics with 2Mbps sending rate each. Therefore, the aggregated cross traffics' rate in first two router is 2Mbps, second pair of routers is 4Mbps and third pair is 6Mbps. Bottleneck is the wired link between third pair and background traffic is stable. Average throughput obtained using TCPRENO, TFRC, UDDJ-TFRC, TCP-Westwood and JTCP are plotted in Figure 5. JTCP and TCP-Westwood make false-estimation in detecting congestion loss in wireless when background traffic make its delay measurement noisy. In addition, TFRC does not show TCP-fairness when wireless loss is slight because background traffic is CBR and congestion is caused by TFRC and TCP-RENO itself. When wireless loss is heavy, the loss event measured by TFRC is close to that measured by TCP-RENO, in which the total loss event is dominated by wireless loss. Under this circumstance, throughput of TFRC is close to that of TCP-RENO. Throughput of JTCP and TCP-Westwood in Figure 6 conduct that their delay measurement will be interfered by bursty background traffic as well as the accuracy of congestion detection is go from bad to worse as channel loss became serious. Compare to JTCP and TCP-Westwood, UDDJ-TFRC can fight against bursty background traffic and interference of wireless loss and utilized bottleneck bandwidth. The results also imply that our congestion prediction can help transport protocol keep in congestion avoidance status to provide better application QoS even if channel loss is not exists.

3.2 Friendliness of UDDJ-TFRC

Another key issue for TCP-like protocol is TCP-friendly in which denotes that every connection share bottleneck fairly. In other word, TCP-friendly congestion control cannot repress throughput of TCP. In this section, we used two topologies to evaluate intra-fairness, inter-fairness and intrusiveness of UDDJ-TFRC.

Intra-fairness

The intra-fairness is defined as the fairness between same protocols. To evaluate intra-fairness, we used Jain's Fairness Index, which is defined as [6]

Original Article

$$\varphi = \frac{(\sum_{i=1}^{n} b_i)^2}{n \times \sum_{i=1}^{n} b_i^2} \quad (13)$$

If φ ≈ 1, then the transport protocol is believed to fairly share bottleneck bandwidth. We evaluated Intra-fairness of UDDJ-TFRC, TCP-Westwood and JTCP in homogeneous topology. In this scenario, all connection share the same bottleneck, which is wireless link. We built multiple connections into the links and evaluated bandwidth utilization and their fairness index. Although three approaches show good intra-fairness, UDDJ-TFRC can achieve better intra-fairness with high bottleneck bandwidth utilization (more than 95%).Simulation shows that TCP cannot full-utilize bottleneck bandwidth even if wireless loss does not exist. (2) UDDJ-TFRC provide better application QoS than other TCP flows by utilizing residual bandwidth without repressing other TCP flows. (3) As the increasing number of UDDJ-TFRC flows, UDDJ-TFRC share bandwidth with other UDDJ-TFRC flows fairly even if background traffic exist. (4) As the increasing the random loss ratio, sending rate of UDDJ-TFRC increased by utilizing degraded throughput of other TCP flows.

Random Loss Ratio (%)

Fig.4 Average throughput comparison in single hop

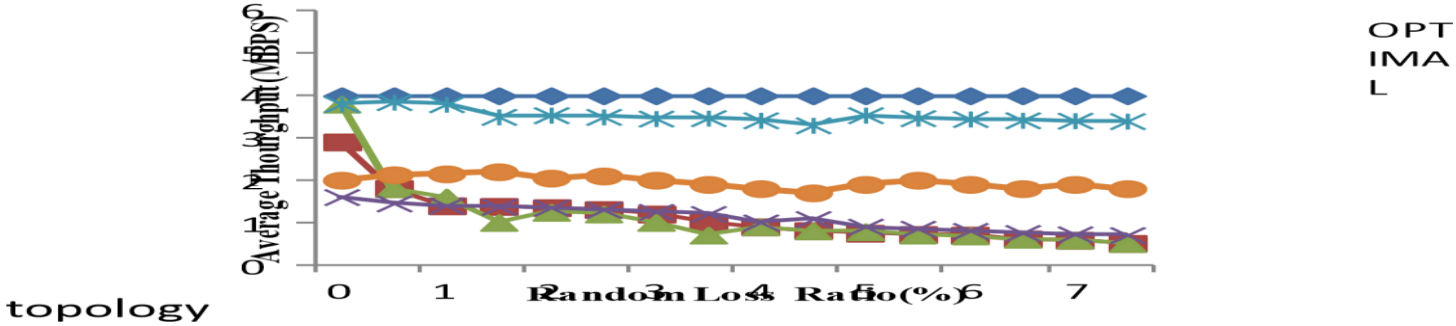


Fig.5 Average throughput comparison in multi hop with path persistent traffic

IV. Conclusion and Future works

To improve application QoS this paper introduces the heuristic design principle of end-to-end wireless transport protocols, and concluded that the common problem of these approaches are the bursty nature of background traffic and intrusiveness problem. In this paper, we presented a modified TFRC protocol, UDDJTFRC, which uses uni directional delay jitter based congestion control to overcome the faulty probelm of TFRC in wired-wireless networks when background traffic is bursty. The performance improvement is established for TFRC using correct congestion signal which is detected by uni directional delay jitter. UDDJ-TFRC achieves end-to-end semantic, intra- and inter-fairness, and wireless adaptability. UDDJ-TFRC has been experimented through several simulations with different topology and background traffic model. The simulation results show that UDDJ-TFRC gains throughput improvement in wired-wireless network over TCP-Westwood, JTCP. In the future, we will continue more experiment analysis in terms of variable packet size, queue length, complex topologies and loss model. We also plan to implement UDDJ-TFRC and conduct real experiments in the next generation network such as WiMAX environments.

Original Article

References

H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz. A comparison of mechanisms for improving tcp performance over wireless links.IEEE/ACM Trans. Netw., 5:756–769, December 1997.

H. Balakrishnan, S. Seshan, and R. H. Katz. Improving reliable transport and handoff performance in cellular wireless networks. Wirel. Netw., 1:469–481, December 1995.

S. Biaz and N. H. Vaidya. ”de-randomizing” congestion losses to improve tcp performance over wired-wireless networks. IEEE/ACM Trans. Netw., 13:596–608, June 2005. [4] S. Cen, P. C. Cosman, and G. M. Voelker. End-to-end differentiation of congestion and wireless losses. IEEE/ACM Trans. Netw., 11:703–717, October 2003.

S. Floyd and K. Fall. Promoting the use of end-to-end congestion control in the internet. IEEE/ACM Trans Netw., 7:458–472, August 1999.

X. Liu, K. Ravindran, B. Liu, and D. Loguinov. Single-hop probing asymptotics in available bandwidth estimation: sample-path analysis. In Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, IMC ’04, pages 300–313, New York, NY, USA, 2004. ACM.

S. Mascolo, C. Casetti, M. Gerla, M. Y. Sanadidi, and R. Wang. Tcp westwood: Bandwidth estimation for enhanced transport over wireless links. In Proceedings of the 7th annual international conference on Mobile computing and networking, MobiCom ’01, pages 287–297, New York, NY, USA, 2001. ACM.

E.-K. Wu and M.-Z. Chen. Jtcp: jitter-based tcp for heterogeneous wireless networks. IEEE Journal on Selected Areas in Communications, 22(4):757 – 766, may 2004.

K. Xu, Y. Tian, and N. Ansari. Tcp-jersey for wireless ip communications. IEEE Journal on Selected Areas in Communications, 22(4):747 – 756, may 2004.

N. Yoma, J. Hood, and C. Busso. A real-time protocol for the internet based on the least mean square algorithm. IEEE Transactions on Multimedia, 6(1):174 – 184, feb. 2004.