Routing Failure Recovery with Contention Window Adaptation in Wireless Networks

Jubilant J Kizhakkethottam
Department of Computer Science
St Joseph's College of Engineering and Technology, Kottayam, Kerala
jubilantjob@gmail.com

S Karthik
Department of Computer Science
SNS College of Technology
Coimbatore, Tamil Nadu
kkarthikraja@yahoo.com

Vinod Chandra S S
Computer Centre
University of Kerala
Thiruvananthapuram, Kerala
vinodchandrass@gmail.com

***Abstract* — Due to contention through wireless shared channels, Real Bandwidth Delay Product (BDP) never reaches a value as big as one in wired network. Overshooting window problem is caused by congestion control algorithms which ultimately give poor throughput in Mobile Networks. The work concentrates to overcome the above mentioned problems. For this, the Round trip Time (RTT) is divided into two parts: a) Congestion RTT and b) Contention RTT. Contention RTT has no direct effect on BDP and BDP is determined by congestion RTT, if a link with worst contention status does not result in link breakage. Insufficient use of contention RTT results in Transmission Control Protocol (TCP) Congestion Window overshooting problem. Our work is to propose a new mechanism called Congestion Window Adaption with Routing failure Prevention that provide better and more accurate method of estimating the contention status. Congestion Window Adaptation-Retransmission Reduction (CWA-RR) is used to overcome routing failure. Based on this mechanism, our method prevents routing failure to limit the window size from overshooting. This is done by establishing connection between Transport layer, Network layer and MAC Layer. Results shows that proposed mechanism out-performs the conventional TCP and enhanced mechanism called TCP with contention control.**

Keywords* — *Bandwidth Delay Product (BDP), Round trip Time (RTT), Contention RTT, Request-to-Send (RTS)

I. INTRODUCTION

A Mobile network can be defined as a network with completely self-organized and self-configuring capabilities requiring no existing infrastructure or administration. Transmission Control Protocol (TCP) is designed to provide reliable end to end delivery of data over unreliable network and outperform conventional wired networks. TCP encounters challenges in mobile networks [1], [2]. Due to instability and shared wireless channels, Mobile networks may suffer from impairments. For example, route failures, drops caused because of Medium Access Control (MAC) contentions and interference, and random channel bit errors. Theoretically, TCP works without consideration of the implementation at lower layers, but the performance of TCP significantly decreases in such unreliable networks. Route failures, which affects the network performance significantly, is considered an important research issue. This problem was examined, which includes optimization of the TCP protocol, taking to consideration route failures in dynamic networks [3]. The work concentrates on the consequence of contention and interference factors in the performance of the network.

Mobile networks adopting IEEE Standard 802.11 MAC layer faces contention and the hidden-node problem. The contention problem in wireless networks happens when multiple adjacent nodes are contending for a shared channel to transmit its packets. Additional problem is the hidden-node terminal problem. When two nodes connect to each other, other nodes within the interference area of these two communicating nodes cannot access the channel. When a node attempts to transmit a packet, it contends with the neighboring nodes that are not adjacent to access the wireless channel. The extended hidden-terminal problem is a representative issue resulting from the mentioned properties [4]. Some nodes may not respond to the Request-to-Send (RTS) from the neighboring node in as it cannot access the channel, which has been employed for communication with some other neighbor. If one RTS sender cannot receive a clear-to-send (CTS) reply within a maximum number of retransmissions (generally 7 times), it consider this situation as a link failure, and data packet the drop results.

Newly designed TCP congestion-control method is to increase the amount of segments up to network capacity, which is designated as bandwidth-delay product (BDP). Chen et al. examined the BDP in both wired and wireless networks [5]. In conventional networks, segments pass through links back to back; ultimately these segments may chock up the whole network pipe. Nevertheless, a receiver should start forwarding a segment after completely receiving it from the sender in multi hop mobile networks. In addition, consider the channel contention and channel interference problem, the BDP of a TCP connection is much smaller in wired networks.

The price of the TCP congestion window (cwnd) is proportional to the BDP, and a congestion-control algorithm in wired networks, tries to keep the value of cwnd value near the BDP. The real BDP in Mobile

networks is much smaller than in wired networks. If we select the original method for controlling the cwnd, which always tends to go beyond the real BDP, the complete network will be overloaded and operates under a bad congestion environment [6]. This situation is an example of a TCP congestion window overshooting problem, which considerably reduces the network performance.

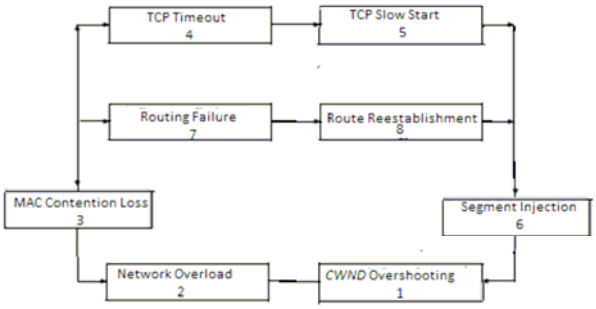

Fig. 1. TCP congestion window (*cwnd*) overshooting problem

Fig. 1 shows a flow of TCP cwnd overshooting problem. During cwnd upgrading and normal data transfer step, cwnd overshooting (phase 1) causes a TCP network to be overloaded soon (phase 2). A large number of data segments need to be transferred and severe MAC contentions may accordingly occur. This results, many segment losses (phase 3), which trigger retransmission timeouts (RTOs; phase 4) and subsequent slow start (phase 5) at the TCP source node. The Data segments are injected into the network (phase 6), at a lower transmission rate. However, an increase in the cwnd will exceed an appropriate level in accordance with the real BDP, and it initiates a return to phase 1. The TCP cwnd value overshoots, resets frequently in the problem cycle, and a large change in the cwnd considerably reduces the throughput. Fu et al. examined the effect of MAC contentions on routing [7]. In Fig. 1, a frame loss occurs due to MAC contention, a routing agent identifies a route failure (phase 7) and intimates the source node to re-establish a route (phase 8). If route is re-established through route discovery, more control packets pass through the network which makes the congestion status even worse.

To solve the TCP congestion window over- shooting problem, a novel mechanism was proposed called congestion window adaptation through Routing Recovery (CWA–RR). The main contributions of the work are listed below: `

1) The Real round-trip time (RTT) remained divided into two parts: a) congestion RTT and b) contention RTT. The contention RTT no influence on the Bandwidth Delay Product (BDP) as BDP resolution is by only the congestion RTT, if a link with the worst contention status does not lead to link breakage. Inadequate use of contention RTT marks TCP congestion window overshooting problem.

2) A variable was defined-Variance of Contention RTT per hop (VCRH) for evaluating the degree of link contentions. Mainly, it represents the degree of link contention. Next, the variance is a random variable which reflects the contention situation observed during the recent observation window. Finally, the variance can reflect the status of a bottleneck.

3) Real round-trip time (RTT) is computed dynamically based on the values of srtt and rttvar collected from the DSR protocol based on Routing request and routing replay.

4) Retransmission rate is reduced by analyzing route failure messages and by recognizing the route failure identified by DSR protocol.

5) Modification in the TCP congestion window adaptation mechanism was done. The CWA–RR adapts the cwnd value based on RTO, acknowledgement (ACK) and the VCRH. It is timely and Fig. 1 shows a flow of TCP cwnd overshooting problem. During cwnd upgrading and normal data transfer step, cwnd overshooting (phase 1) causes a TCP network to be overloaded soon (phase 2). A large number of data segments need to be transferred and severe MAC contentions may accordingly occur. This results, many segment losses (phase 3), which trigger retransmission timeouts (RTOs; phase 4) and subsequent slow start (phase 5) at the TCP source node. In the subsequent stage, Data segments were injected to the network (phase 6), at lower transmission rate. However, an increase in the cwnd will exceed an appropriate level in accordance with the real BDP, and it initiates a return to phase 1. The TCP cwnd value overshoots, resets frequently in the problem cycle, and a large change in the cwnd considerably reduces the throughput. Fu et al. examined the effect of MAC contentions on routing [7]. In Fig. 1, a frame loss occurs due to MAC contention, a routing agent identifies a route failure (phase 7) and intimates the source node to reestablish a route (phase 8). If route is reestablished through route discovery, more control packets pass through the network which makes the congestion status even worse.

II. CONGESTION WINDOW OVERSHOOTING

The illustration of congestion window overshooting problem for RTT and BDP is shown in fig 2, see end of paper after references section.

A. Congestion Window Overshooting Problem

Examine the Equation 1described below which identify the Upper bound of Bandwidth Delay Product (BDP_UB) in Mobile networks.

$$BDP\_UB = S \times (\sum_{i=0}^{m} df + \sum_{i=0}^{n} dr) / (4d_{max}) \quad (1)$$

In the equation, S is the size of segment. The remaining numerator is the sum of per-hop delays through forward path (df) and return path (dr) which indicates RTT, and it shows that the BDP is proportional to path length. The dmax is the maximum per hop delay among df and dr , and it proves that the BDP is tightened by the bottleneck link, that suffers from the nastiest contention. The coefficient 1/4 is resolute from a simplified but representative chain node model in which the maximum spatial reuse is 1/4. The derivation is an ideal upper bound of the BDP.

BDP is a key factor in shaping the transmission rate of segments, and is obtained through the direct measurement of the network status. Consider a network pipe, the width of the pipe represents the bandwidth at a bottleneck and the length represents RTT. The BDP is a product of the bottleneck bandwidth and RTT, and is given in (1), regardless of the coefficient 1/4. In conventional networks, packets constantly flow through the pipe. In Mobile networks with link contention, a node should access a channel before sending data segments. It needs to suspend transmission for a period of contention delay at each link and data segments cannot constantly flow through the network pipe. If we reduce the halt time of flow by leaving only the time of continuous data flow, the approach will be similar to the conventional BDP. In mobile networks, the BDP is determined by the time of continuous flow of data segments.

Based on the above result, the real RTT is spitted into two parts—congestion RTT and contention RTT—as in Fig. 2. The contention RTT is the sum of contention delays through the path, which is a precise but essential factor in mobile networks. Conversely, the remaining part in RTT is the congestion RTT, made up of the end to end transfer delay of all the links through the path. It is to be considered that this end-to-end transfer delay of link contains queuing delay, transmission delay, and processing delay but no contention delay, and we will call it the link transfer delay in this paper. The duration of continuous segment flow in the pipe is determined by congestion RTT. BDP is determined by the congestion RTT and not by the contention RTT. However, in a path with severe contention, total transmission rate of segment flow reduces to a large extent, resulting in decreased throughput. The contention RTT is unconnected to BDP, and there are no bottleneck effects in the contention RTT. In other words, a contention in a path is not determined by a link with the worst contention status. However, the conclusion has an important prerequisite that the link with the worst contention status does not lead to link breakage. Nevertheless, the bottleneck link with the worst contention status should not totally be neglected, as the contention becomes worse at an intermediate node, there is a higher possibility of a link failure.

In order to examine the TCP congestion window overshooting problem, an increase in the congestion window allows a raise in the amount of segments up to the level of BDP. If the original congestion-control method [6] is directly applied over 802.11, the TCP congestion window is adjusted according to the real RTT (directly measured at the TCP agent). However, the congestion RTT, which is associated to the network pipe volume, is smaller than the real RTT. Therefore, the congestion window is likely to surpass the level of the BDP. In addition, the higher the ratio of the contention RTT to the congestion RTT is, the greater the overshooting problem.

Triantafyllidou et al. proposed a correlated method and introduced a time-based expression for the BDP, which is used for dynamic adaptation of the value of CWNDmax [14].

$$CWNDmax = \frac{packet_{length} \times RTTTCP}{2(n-1) \times MAX\{di\}} \quad (2)$$

In (2), is delay at node i considering wired networks, BDP is the product of bandwidth and RTT. It represents load capacity of the link. Conversely, in wireless networks, load capacity depends on a number of factors, (e.g., transmit power, node density, and buffer size). Thus, in real networks, owing to complex contention situations, it is hard to measure the real BDP.

B. Understanding the BDP Equation

Based on the aforementioned insight, a modified equation of BDP is presented as follows:

$$BDP = (S/Dcongestionmax) \times RTT_{congestion} \quad (3)$$

where *Dcongestionmax* is the maximum link transfer delay.

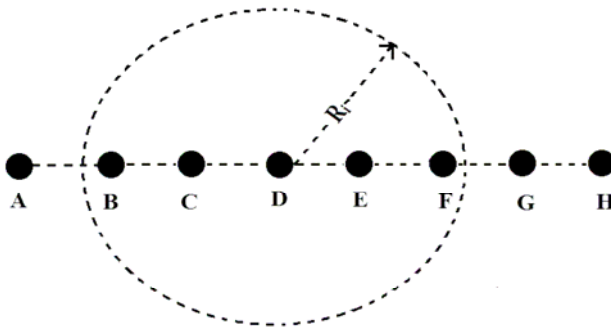

Fig 3 . Typical Chain Topology

Fig. 3 shows the network scenario using a typical chain topology, where the transmission and interference ranges are 250m and 550 m, respectively. In these circumstances, when node D receives packets from node C, all nodes that remain in the range of interference (Ri) of node D should be blocked. While the CD link is under transmission, the DE, EF, and FG links are under contention. Assume that the transfer delay per hop is the same for all links; so the contention delay is three times longer than the link transfer delay. Through the whole path, the congestion RTT is one fourth of the real RTT.

The coefficient 1/4 in (1) is derived from the aforesaid analysis result. Still, the conclusion about the bandwidth in (1) is different. Dmax is the per-hop transfer delay, that includes the link transfer delay, the contention delay, and thus, S/dmax is the rate of packet sent from a node to the next hop, in view of the contention delay. The remaining terms 1/4 RTTreal correspond to the link transfer delay, which is the irrationality in (1). In addition, (1) is restricted in the abovementioned chain topology, and the modified BDP equation (3) is more general.

III. CONGESTION WINDOW ADAPTATION WITH ROUTE FAILURE RECOVERY (CWA–RR)

A new mechanism was recommended in the called Routing Failure Recovery with Contention Window Adaptation (CWA–RR). This section introduces a new parameter called the VCRH and explains the reason that this parameter can be used to estimate the degree of MAC contentions. Based on this parameter, we improve the TCP congestion window adaptation mechanism to solve the overshooting problem.

1) VCRH

To define a parameter called the VCRH, first, the value of the contention RTT is gathered through packet forwarding. In the MAC agent, the contention delay is calculated as the interval between a packet arrival at the head of line of the queue and the delivery time to the physical layer. A time stamp was assigned in the fragment (frame) to keep track of the contention delay of the fragment in the current node. The contention delay is added hop by hop and recorded in the fragment, and finally, the TCP destination copies the sum from the segment into the ACK. The two points are noted here:
- The delayed ACK function should be disabled and
- The fragmentation function in simulation is disabled to simplify the fragment sampling of the contention delay [20].

As a result, the MAC agent could copy the sum of contention delay straightaway from fragment to the packets. During real implementation, MAC agent chooses the contention sample in the last fragment during the fragmentation phase and copies it from the packet. Sample value of the contention delay is additionally recorded in the ACK, and TCP source obtains the contention RTT from the newly received ACK. Divergence of the contention RTT of the latest n segments can be figured out at the source node, and is called as variance of contention RTT (VCR).Last, the VCRH is projected as VCR divided by the number of hops in a given route.

There are three reasons to adopt the VCR to calculate degree of link contentions. 1) It can represent the degree of link contention. If the possibility of channel contentions is larger, the MAC contention window needs to be larger, and the back off time is selected in a wider range; hence, a higher contention probability results in a larger variation of the contention delay. We calculate this jittery through variance, and it could directly reflect the contention possibility. 2) The variance is a random variable that reflects the contention situation observed during the recent observation window. 3) The variance can also reflect the status of a bottleneck. Assume that there is a bottleneck in one of the two paths; and its bottleneck has a larger VCR. If there is a bottleneck link due to rigorous contentions, there is a high probability of link breakage.

In this method, there are three details to be explained. First, we remove the effect of path length. After obtaining the contention RTT, we can calculate the variance and divide it by the number of hops. Then, obtain the deviation for each hop. Second, consider a round-trip path instead of a forward path used in the conventional method [15], because ACKs in the backward route are also an important factor for detecting link contention. Third, we need to set an appropriate value of n, i.e., the required number of samples for calculating the VCR. If n is very large, it increases the memory size and calculation cost of nodes. However, it gradually responds to a change due to a long-term average effect. Conversely, if n is very small, it is very sensitive to a small change and may cause a large value of VCRH on different RTTs. During implementation, we use the following recursive equations to update the VCR once the source receives a new ACK.

See equations 4, 5, 6 and 7 in the appendix at the end of the paper after the references section.

Here, ACK is the sampling chain of newly received ACKs, Nmax is the maximum number of samples, EACK and VACK are the anticipation and the variance of CRTT (contention RTT) from the latest ACK (when ACK < Nmax) or Nmax (when ACK >= Nmax) ACKs. CRTTnew is the anew obtained CRTT value and CRTTold is the old CRTT value. If ACK < Nmax, VACK is obtained from (4) and (5), because the number of sample entries ACK is different from Nmax . These two equations are used during the short transient time right after the TCP connection initialization or the timeout. In

this case, CRTTold is not needed. If ACK >= Nmax, VACK is obtained from (6) and (7). In this case, a new CRTT sampling value contained in a new ACK pushes out an old entry Nmax times ago, and we can update the VCR according to the new CR value itself. When VACK is newly obtained, we can obtain the VCRH by dividing VACK by the number of hops.

2) TCP Congestion Window Adaptation Mechanism

A threshold parameter VCRH_th was set, and if the VCRH exceeds this threshold, the degree of contention in the network is considered to be severe. When a TCP source receives a new ACK, it updates value of the VCRH before examining whether this value is larger than or equal to the VCRH_th value. If it is found larger, cwnd is decreased by one MSS instead of an increase in the congestion window. This approach could effectively restrain the congestion window from overshooting. If RTO expires, which indicates that the network is in a bad congestion or contention status, we need to confirm whether the VCRH is larger than or equal to VCRH_th before having the congestion window threshold (ssthresh) [6]. If the VCRH is larger, the network condition is considered to be very bad, and we reset the cwnd to 2*MSS and enable the slow-start step. Otherwise, the contention status of the network is not so bad, and we just halve the cwnd. We should notice that the VCRH sampling should be reset after an RTO, and thus, the subsequent slow start of cwnd is not affected by the low value of VCRH obtained before the RTO expires.

The new technique regards a large change of contention RTT as a proof of contention status in contrast with previously mentioned [15]. This mechanism is more accurate, as it estimates the contention from a series of contention RTT values. The method is timelier, because it eradicates the outdated values long before, but in the conventional TCP contention control, the outmoded contention information affects the computation of VCRH. On the other hand, our congestion window adaptation mechanism is simpler and steadier. First, upon receiving a new ACK, the VCRH result indicates how we can regulate the value of cwnd, and even if the evaluation on contention is different from reality, the resultant cwnd is just slightly modified. However, in the conservative TCP contention control, once the judgment on throughput and contention is worse, the cwnd is reset. Subsequently, if a packet loss occurs, it determines whether we can minimize the cwnd, depending on the contention situation. If the contention status is not severe, it is helpful for alleviating the large dissimilarity of cwnd and increasing the throughput by just halving the congestion window. Our new mechanism is more robust and more accurate.

3) RTT Computation Based on DSR (Dynamic Source Routing Protocol)

It is the function of the network layer to select the shortest route [26]. The Transmission of data through this route is obligation of transport layer [27]. The after the selection of shortest route by network layer, if link failure occurs due to mobility in this route, it is uninformed to MAC layer and Transport layer. This results in retransmission of data packets without knowing link failure. The ultimately results in the following a) high packet loss, b) increased delay, c) reduced data delivery etc. These Link failure problems can be solved if the both layers are linked with network layer. This linking with the layers makes the information transfer between the layers.

As 802.11 are designed to supply reliable service at the link level, it provides re-transmissions of the same packet until the retry limit is reached. Apart from the delays like back off delay, contention delay, and delay discussed before, there is additional latency due to the MAC re-transmissions. Hence there is large variation in the time an ACK packet arrives at the sender. This translates to imprecise estimate of round-trip time at the sender resulting in false re-transmissions or large estimates of Retransmission timeout (rto) both of which have an unfavourable effect on TCP performance. The aim is estimating the round-trip time more precisely with the help of inputs from the routing layer concerning the network path characteristics.

For each connection, TCP preserves a measure of average round-trip time in smoothed round-trip time (srtt) and variation in rtt in rtt variation (rttvar). It estimates the round-trip timeout (rto) of a packet using the following formulate. This value of rto is retained for a number of packets until a new estimation is done. The variable rtt gives the current round-trip time measurement.

$$rtt=(1-\alpha)srtt + \alpha.rtt \quad (8)$$

$$rttvar = (1-\beta)rttvar + \beta.|rtt - srtt| \quad (9)$$

$$rto= srtt + 4.rttvar \quad (10)$$

where α and β symbolize the smoothing factor for the updates. At the start of the connection rto is taken as3 (varies with implementation). The first measurement of srtt is taken as rtt and rttvar as rtt/2. The true value of rto arrives after a number of such measurements are made. When a packet is re-transmitted, updating rto based on these measurements leads to a problem called 'retransmission ambiguity' addressed in [8]. This method of computing round-trip time is workings well with wide-area Internet. But in ad-hoc networks using 802.11 which

have high and varying delay and that cause frequent ROUTE-FAILUREs, this method of evaluation does not lead to fast convergence of true rto value. The problem of rto convergence and divergence is aggravated in such networks. False convergence leads to forged re-transmissions and divergence leads to delay in loss detection which is wasteful of resources. The work uses the input from the network layer regarding the network path characteristics to arrive at a better estimate of round-trip time. DSR which is the routing layer in our case helps in this. DSR uses ROUTE-REQUEST packets to discover a source route for a packet.

The information concerning the number of hops and round-trip time measured at DSR is passed to TCP. With this information obtained from DSR, we maintain different estimates of rto for different number of hops. Since we integrate hop based information for timer estimation in our protocol we call it HTCP. At the commencement of the connection, or for the first measurement for that hop, srtt for that entry is taken as time difference between sending of ROUTE-REQUEST packet and arrival of ROUTE-REPLY packet of DSR and rttvar as srtt/2. As the connection progresses, different sets of values for srtt, rttvar and rto are maintained for diverse routes. If a route change happens due to route failure, DSR informs that to the TCP sender. DSR knows about it through ROUTE-

FAILURE messages it gets from the network. This is informed to the TCP source which keeps away from re-transmitting the packet.

Retransmission is a problem both from the network viewpoint where contention increases and sender viewpoint where it leads to incorrect updates of rto. Hence preventing re-transmissions do well to improving performance. The achievement by this approach is three fold. Maintaining different values of rto for different routes prevents the problem of random variations in rto estimations which could result from regularly varying routes and recurrent route failures. This helps in better calculation approximately for that specific route or the route using particular number of hops. In standard TCP no value of srtt is allocated at the start of the connection. In our work we obtain this value from DSR. Since we take rttvar as srtt/2, a variation in this value is accepted. This leads to quicker convergence to true value of rto. Informing TCP sender about ROUTE-FAILURE helps in preventing false/spurious transmissions.

IV. RESULTS AND DISCUSSION

The enactment of original TCP, the conventional TCP with contention control mechanism [15] and our proposed mechanism – Routing Failure Recovery using Contention Detection [21] are compared. Focused on the effects of disagreements and intervention to reduce the effect of route failures as much as possible, attention was on grid topology as static state. The work also examined the network performance and characterized the congestion window among the three mechanisms. For this, we also focus and present results in a dynamic random topology.

A. Grid Topology

The Fig. 4 shows structure of static grid network as the testing topology containing m2 nodes. In a static situation without mobility, exploration can be done based on the performance of contentions in the grid network. The space among two end-to-end nodes is set to be 150 m, the transmission is set to 250 and interference radii is set to 550 m, consistently. Different scale of grid networks is generated, with m incremented from 4 to 9 systematically.

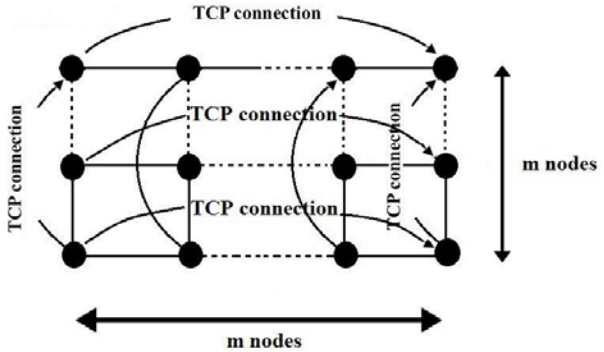

Fig 4: Grid Topology

There are 2m connections in the grid; this topology causes a large number of contentions among these connections. In every row, a TCP connection is established from the left end node to the right end. In each column, a TCP connection is expected to get established from the bottom end node to the top end node. These TCP connections are of the same parameters, i.e., a segment size of 512 Bytes and a maximum window size of 32. The connections are chosen randomly and each data point in the figures takes the probability of ten sample values.

TABLE I.
EXPERIMENT PARAMETERS IN GRID TOPOLOGY

Topology Pattern	**Grid /Static**
Distance to Adjacent node	**150 m**
Transmission Radius	250 m
Interference Radius	550 m
Segment Size	512 bytes
Maximum window size	32
Simulation Time	500 s
Number of nodes	16,25,36,49,64,81
Number of connections	8,10,12,14,16,18
Number of VCRH samples	6 segments
Routing protocol	DSR

Features of TCP congestion window among the three mechanisms are scrutinised: 1) the original TCP; 2) TCP contention control; and 3) TCP with CWA–RR. The experiment parameters are set according to Table I. First, we compare our proposed TCP CWA–RR mechanism with the original TCP, it is found that the proposed mechanism effectively decreases the times of cwnd reset. That means, the regular reset problem shown in the original cwnd is reduced, and this is clearly shown in the result. The proposed mechanism reaches the competence of the network pipe after a period of cwnd increases along with the conventional TCP, but later, the cwnd of the normal TCP still goes up aggressively. The TCP contention control works are investigated. The cwnd value of this mechanism is continuously very small and widely varies.

This shows that this method can indeed confine the cwnd values from excessive increase, but this adaptation mechanism privations a stable estimation on the contention status. It directly maps the fluctuant round-trip contention information to the adaptation of a new cwnd value, and thus, the adaptation is quickly switched between an increase and a decrease in the cwnd values. Third, in the TCP with maximum window mechanism, an increase in the cwnd value is slow, but the cwnd values frequently vary, because RTT TCP is not the real RTT and could not reflect the actual network state. In Fig. 6 cwnd value of the TCP with CWA–RR is more stable. When receiving a new ACK, the node adjusts the cwnd value based on the VCRH value. Although the estimation on the contention is different from reality, the resulting cwnd is just slightly modified. If a segment loss occurs, we determine whether we can minimize the cwnd, depending on the contention situation. If the contention status is severe, it is beneficial for alleviating the large variation of cwnd and increasing the throughput by just halving the congestion window. The graphs indicate the Comparison in normal TCP, when Contention is taken and when Routing Failure Prevention is implemented, our proposed work in various five parameters.

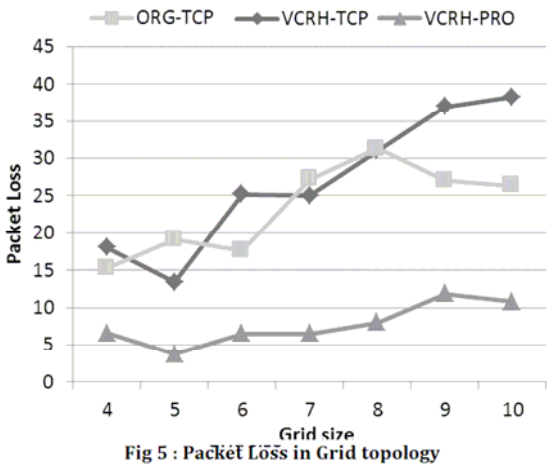

Fig 5 : Packet Loss in Grid topology

I. Packet Loss: Packet loss is the failure of one or more transmitted packets to arrive at their destination. Packet loss is distinguished as one of the three main error types encountered in digital communications; the other two being bit error and spurious packets caused due to noise. As indicated in figure as the grid size improves the rate of packet loss increases. Packet losses in congestion control are inevitable owing to uncertainties and highly time-varying traffic patterns in the best effort service model [22] or in conventional end-to-end rate control scheme which uses additive-increase multiplicative decrease (AIMD) [23]. Furthermore, congestion problems result from a mismatch of offered load and available link bandwidth between network nodes. Such problems can cause high packet loss rate (PLR) and long delays, and can even break down the entire system because of congestion collapse. In other words, when the load exceeds the network capacity, any increase in the offered load leads to a decrease in the useful work done by the network. The main task of the congestion control methods is to utilize the buffers of the controlled load service as optimally as possible. The data rates of controlled data flows should be kept as high as possible without losing packets during controlled data flows. The rate of loss of packet is increasing based on the increase in grid size, in original TCP and VCRH TCP but very less in our proposed algorithm and remains fairly constant.

II. Normalized Routing Load: Normalized Routing Load (or Normalized Routing Overhead) is defined as the total number of routing packet transmitted per data packet delivered at the destination. Each hop-wise transmission of a routing packet is counted as one transmission in NRL. It increases as the number of nodes in the grid increases. The rate of growth is very less in proposed algorithm compared to original TCP and VCRH TCP as indicated in the fig. 6.

III. Packet Delivery Fraction (PDR): Packet delivery ratio is the ratio of the number of delivered data packet to the destination. This illustrates the level of delivered data to the destination. The greater value of packet delivery ratio means the better performance of the protocol. While the packet delivery ratio is very high in proposed work with almost constant nearing 100 % consistently, both the other techniques, decrease in PDR was decreasing as the number of nodes increased.

IV. Throughput: Throughput is the average rate of successful message delivery over a communication channel. This data may be delivered over a physical or logical link, or pass through a certain network node. Increased rate of throughput indicates the higher efficiency. From the figure we understood the rate of decrease in throughput as the grid size increases. A constant decrease is exhibited by the proposed algorithm whereas both the others shows a bigger decreasing rate.

V. Average Delay: The average delay is calculated as the sum of time delays that a packet experiences a) deferring at all back off stages, b) at packet's unsuccessful transmissions, and c) final packet's successful transmission [24]. Efficiency is highest if the delay is minimum. Assessing from the graph, we can understand the Original TCP exhibits highest delay and proposed algorithm displays lowest delay.

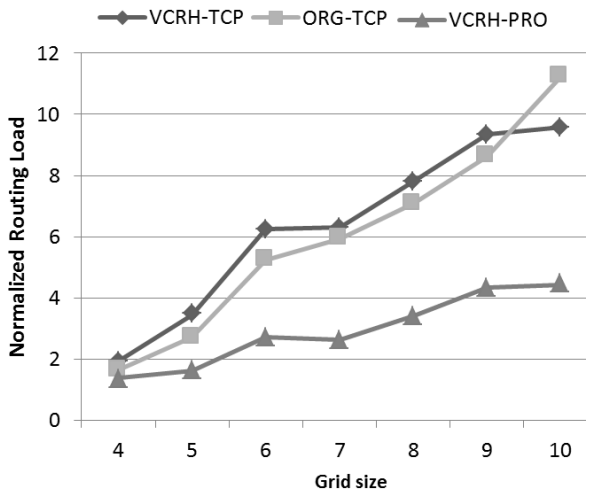

Fig 6 : Normalized Routing Load in Grid topology

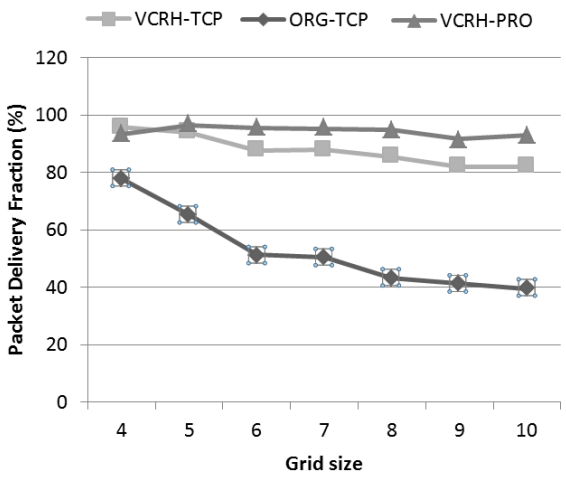

Fig 7 : Packet Delivery Fraction in Grid Topology

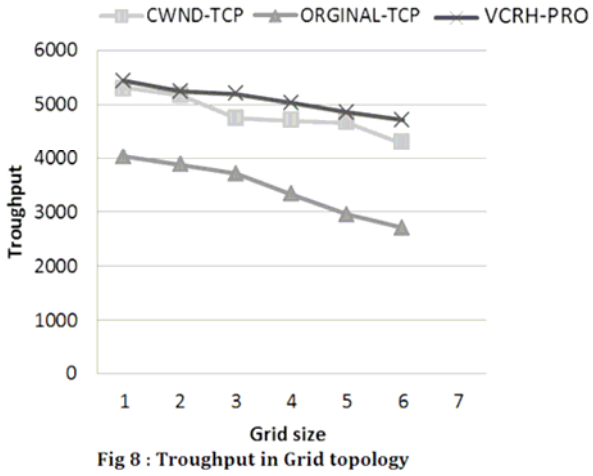

Fig 8 : Troughput in Grid topology

Fig. 10: see Appendix at the end of paper after the references section.

A. Dynamically Random Topology

TABLE II
EXPERIMENT PARAMETERS IN GRID TOPOLOGY

Mobility Pattern	**Random Waypoint**
Area	**500 m * 500 m**
Distance to Adjacent node	150 m
Transmission Radius	250 m
Interference Radius	550 m
Segment Size	512 bytes
Maximum window size	32
Simulation Time	500 s
Number of nodes	50
Number of connections	50
Number of VCRH samples	6 segments
Routing protocol	DSR

The paper consider conventional random way point mobility model in our experiment topology. Here, a node randomly picks up a locality inside an area and starts to move toward the destination with a random speed that ranges from zero to maxspeed. As the node reaches the terminus, it stops for 1 s. The parameter maxspeed values can be varied with 5, 10, 15, 20,25 and 30 m/s in different experiments. We initialize a set of 50 nodes that were randomly distributed in an area of 500 m by 500 m. There are 50 TCP connections between randomly selected sources and destinations. In a static value of speed, paper randomly set up various network scenarios as experiment samples and obtain various data after taking the probability of ten sample values, respectively. In total, there are 50 different network scenarios for varying node speeds. Table II lists the simulation parameters. The Graphs below shows the comparison of Packet Delivery Ratio, Normalized Routing Load, Packet Loss Rate, Average Delay and Throughput of the system.

As indicated in the figure 11, the Packet Delivery Ratio is relatively constant compared to Normal TCP and TCP with contention control mechanism. PDR remains approximately near 100% for the proposed mechanism even when the speed varies from 0 to 30 m/s. In the same conditions, in normal TCP, PDR drops from 80 % to 40 % indicating lower efficiency. In conventional mechanism too, PDR remains relatively low with the proposed method.

In Dynamically Random topology, Normalised Routing Load is very low in the proposed method increasing from 1 to 4 as the speed increases from 0 to 30 m/s as shown in figure 12. In the other mechanisms NRL is considerably high ranging from 2 to 11 for Original TCP and 2 to 10 for conventional mechanism in similar condition. As NRL is considerably low in the proposed mechanism, the efficiency of data transmission is high.

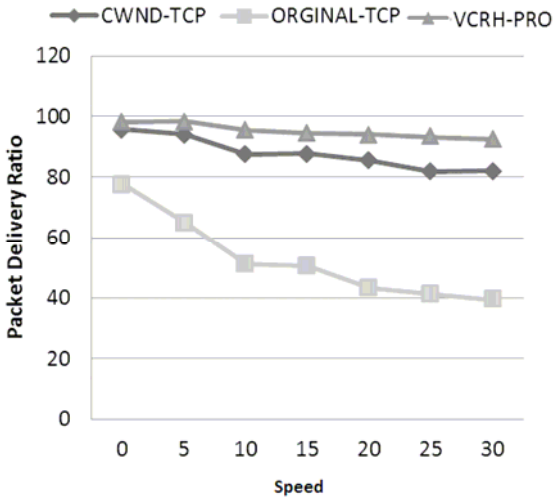

Fig 11 : Packet Delivery Ratio in Random Topology

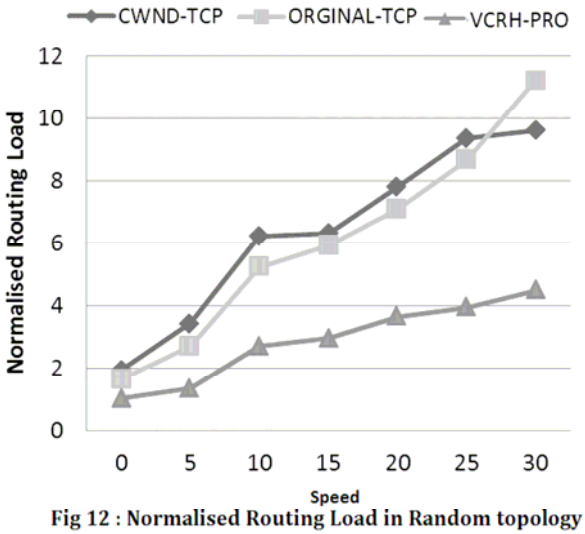

Fig 12 : Normalised Routing Load in Random topology

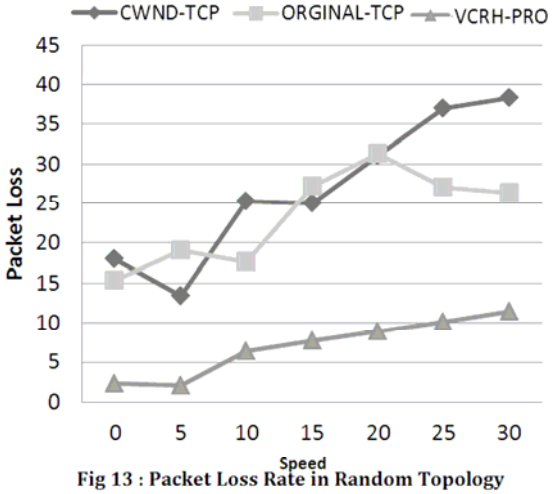

Fig 13 : Packet Loss Rate in Random Topology

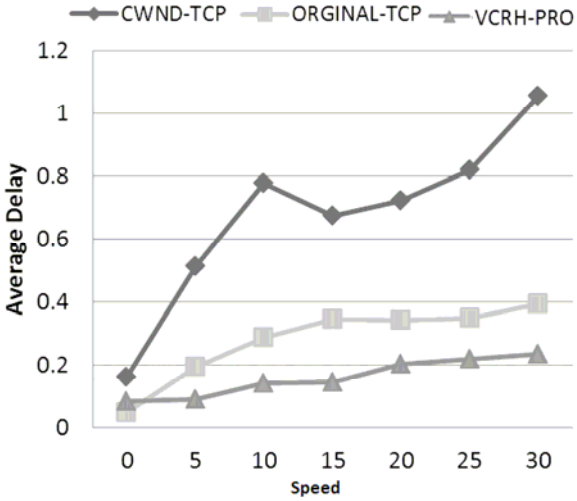

Fig 14 : Averge Delay in Random Topology

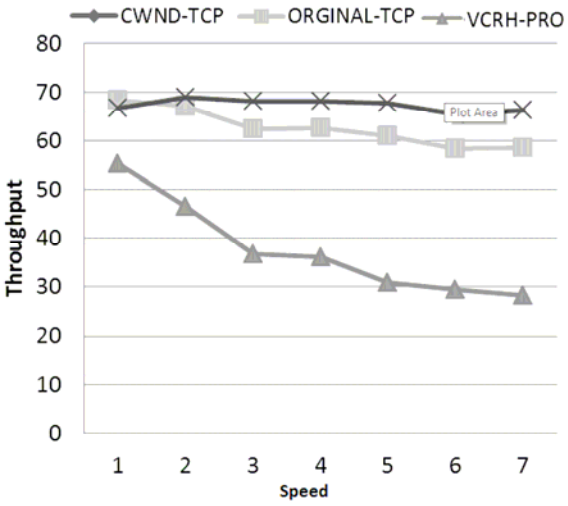

Fig 15 : Throughput graph of random topology

The Figure 13 indicates that as in NRL, the Packet Loss Rate is also significantly low in proposed method stretching from 1 to 10 packets when the other two methods fluctuate to 15 to 40.In all the cases, the seed varies from 0 to 30. The QOS improvement indicated from the graph indicate the performance improvements in the proposed mechanism.

The most considered QOS parameter is Average Delay. From the graph shown in Figure 14, in original TCP when delay ranges from 0.2 to 1 second and in conventional method from 0.1 to 0.4 as the speed upsurges, in the proposed method, the maximum delay displayed was only 0.2 seconds. This low rate of delay indicates the improvement in performance.

The successful rate of packet delivery is indication of throughput as in Figure 15. In Original TCP, when the throughput rate decreases drastically, the conventional TCP shows a decrease in the same. The figure clearly indicates the constant throughput delivery by the proposed mechanism. This result is the indication of extraordinary performance of the proposed mechanism.

The values are taken for normal TCP, TCP using CWND and our proposed work and graph was constructed .The proposed TCP CWA–RR mechanism maximum, when compared with the TCP contention control, and the original TCP mechanisms. This result also shows more improvement than the case in the previous static scenario, and thus, it illustrates that the alleviation of congestion window overshooting still effectively works in the dynamic scenario. However, mobility also plays an important role, except for the contention factor in the dynamic environment, and our new method focuses on contentions and does not consider handling the link breakage from mobility. The development is outstanding, as during link breakdown from mobility, fewer packets are dropped as the congestion window becomes smaller. Thus, shorter end-to-end delays are achieved on the average.

Fig. 10 illustrates the CWND performance of the three mechanisms in the random dynamic topology. The proposed TCP mechanism with CWA–RR gains approximately 6.8% and 13.9% improvements compared with the TCP with maximum window and the TCP contention control mechanisms, respectively, whereas it shows similar performance to the original TCP mechanism. In dynamic scenarios, link breakdowns may frequently occur because of link mobility, and the shrinking of the congestion window is useless in this case.

Based on the results of the experiment, we can observe that the proposed mechanism attempts to fix the adverse effect of link contentions in mobile networks. Monitoring the link contention status, we can effectively alleviate the TCP congestion window from overshooting and keep it in an effective range. The results in the previous static grid topology prove the effectiveness of the proposed CWA–RR mechanism. We split the real RTT into two parts, i.e., 1) the congestion RTT and 2) the contention RTT, and we define a variable VCRH to evaluate the degree of link contentions. This approach is more precise than the CD (contention delay) in the TCP contention control. The CWA–RR mechanism adapts the cwnd value based on not only RTO and ACK but on the V CRH as well.

It is timelier and more accurate than TCP mechanism with maximum window, and it is effective in limiting the congestion window size from overshooting. On the other hand, in the dynamic scenario, because node mobility is another important factor in contrast with contentions, controlling the congestion window cannot totally fix packet losses.

V. CONCLUSION

When the conventional TCP congestion-control mechanism was utilized on top of the MAC 802.11 protocol in mobile networks, the update of cwnd is so aggressive that it soon exceeds an appropriate value according to the real BDP. It may result in a severe contention, segment losses, and a restart of the congestion window. In this situation, TCP connections cannot stably work and yield very low throughput, which has been a major factor for reducing the network performance. Previous studies have given some modified TCP mechanisms. However, there is still space to improve the TCP performance. In our proposed CWA–RR mechanism, we have split the real RTT into two parts—the congestion RTT and the contention RTT—and revealed that the contention RTT has nothing to do with the BDP and that the BDP is determined by only the congestion RTT. We obtained the contention RTT information from forward data and backward ACK at the MAC layer and estimated the contention status through

the deviation of the contention RTT. Then, we made a modification of the window adaptation mechanism, in which the cwnd value was changed according to several factors such as the V CRH and timeout.

The deviation of the contention RTT reflects the direct indication of the contention situation, and this mechanism is timelier and more precise. Furthermore, the relevant congestion window adaptation mechanism is simple and steady, and it does not blindly reset the cwnd but adjust the cwnd value based on the contention status. The experiment shows that the proposed mechanism is more precise and much stable, and it significantly improves the throughput in static scenarios. We have considered an overshooting problem of TCP within the network in this paper. In addition, this mechanism can be used in the connection of the Mobile network with the normal Internet, on the condition that any node with contention in the connection uses our mechanism to contribute the measurement of the VCR and the receiver feedbacks the recorded VCRH. We have used a segment-driven method to sample and calculate the VCRH, which is easy to implement. However, a small problem occurs when the amount of segments is small.

In this case, the time span between segments is large, and thus, some segments as part of the VCRH calculation may be outmoded to some extent. This problem degrades the accuracy in detecting the contention. We can choose another timestamp driven method to alleviate this problem. In other words, every time a new segment arrives, the timestamp of the old segments is checked to determine whether we can remove some old segments before calculating the VCRH. However, the latter method is more complicated. We can combine the packet- and

Timestamp-driven methods to improve the stability of cwnd updating for further work. In addition, we will investigate the adaptation of CWA–RR in other well-known TCP versions.

Our proposed mechanism focuses on link contentions and does not consider the mobility factor. Link failures due to mobility are part of the main sources of link unreliability in mobile Mobile networks (MANETs), which we have previously investigated. In our future work, we will give a comprehensive consideration of these factors to improve the performance of the network.

REFERENCE

[1] Kitae Nahm , Ahmed Helmy and C. Jay Kuo, “TCP over Multihop 802.11 Networks: Issues and Performance Enhancement,” MobiHoc’05, May 25–27, 2005, Urbana-Champaign, Illinois, USA pp. 275–288.

[2] Z. Fu, X. Meng, and S. Lu, “How bad TCP can perform in mobile Networks,” in Proc. IEEE ISCC, 2002, pp. 298–303.

[3] W. B. Zhu, X. M. Zhang, and N. N. Li, “Improve TCP performance with Link-aware warning method in mobile Mobile networks,” in Proceeding. IEEEWICOM, 2008, pp. 1–4.

[4] J. Li, C. Blake, D. D. Couto, H. Lee, and R. Morris, “Capacity of Mobile wireless networks,” in Proceeding. ACM Mobile Communication, Jul. 2001, pp. 61–69.

[5] K. Chen, Y. Xue, S. H. Shah, and K. Nahrstedt, “On-demand multipath distance vector routing in ad hoc networks” Computer. Communication vol. 27, no. 10, pp. 923–934, Jun. 2004.

[6] V. Jacobson, “Congestion avoidance and control,” Computer Commn. Review vol. 18, no. 4, Aug. 1988,pp. 314–329.

[7] Z. Fu, H. Luo, P. Zerfos, S. Lu, L. Zhang, and M. Gerla, “The impact of multi-hop wireless channel on TCP throughput and loss,” in Proceeding. IEEEINFOCOM, 2003, pp. 1744–1753.

[8] C. Y. Luo, N. Komuro, K. Takahashi, and T. Tsuboi, “PACED TCP: A dynamic bandwidth probe TCP with pacing in ad hoc networks,” in Proceeding IEEE Pers., Indoor, Mobile Radio Communication 2007, pp. 1–5.

[9] S. M. Elrakabawy, A. Klemm, and C. Lindemann, “TCP with adaptive pacing for multi-hop wireless networks,” in Proceeding.6th ACM International Symposium Mobile Ad hoc Network Computing, 2005, pp. 288–299.

[10] M. Shen and D. M. Zhao, “Opportunistic link scheduling for multi-hop wireless networks,” IEEE Transaction Wireless Communication., vol. 8, no. 1, Jan. 2009 , pp. 234–244.

[11] Z. Ye, S. V. Krishnamurthy, and S. K. Tripathi, “Use of congestion aware routing to spatially separate TCP connections in wireless ad hoc Networks,” in Proc. 1st IEEE Intn. Conference Mobile Ad hoc Sensor Systems., 2004,pp. 389–397.

[12] H. Zhai, X. Chen, and Y. Fang, “Alleviating intra flow and interflow contentions for reliable service in mobile ad hoc networks,” in Proceeding IEEE Mobile Communication Conference, 2004, pp. 1640–1646.

[13] S. Rangwala, A. Jindal, K. Y. Jang, K. Psounis, and R. Govindan, “Understanding Congestion control in multi-hop wireless mesh networks,” in Proceeding 14th ACM Intn. Conference Mobile Computer Networks” 2008, pp. 291–302.

[14] D. Triantafyllidou, K. Al Agha, and V. A. Siris, “Adaptive setting of TCP’s maximum window in ad hoc multi-hop networks with a single flow,” in Proceeding IEEE WCNC, 2009, pp. 1–6.

[15] E. Hamadani and V. Rakocevic, “TCP contention control: A cross-layer approach to improve TCP performance in multi-hop ad hoc networks,” in Proceeding 5th Intn. Conference on Wired/Wireless Internet Commn.., 2007, pp. 1–16.

[16] X. M. Zhang, N. N. Li, W. B. Zhu, and D. K. Sung, “TCP transmission rate control mechanism based on channel utilization and contention ratio in ad hoc networks,” IEEE Communication , vol. 13, no. 4, Apr. 2009, pp. 280–282.

[17] K. Nahm, A. Helmy, and C. J. Kuo, “Cross-layer interaction of TCP and ad hoc routing protocols in multi-hop IEEE 802.11 networks,” IEEE Transaction on Mobile Computing, vol. 7, no. 4, Apr. 2008, pp. 458–469.

[18] S. Papanastasiou and M. Ould-Khaoua, “TCP congestion window evolution and spatial reuse in MANETs,” Wireless Communication Mobile Computing .vol. 4, no. 6, Sep. 2004, pp. 669–682.

[19] K. Nahm, A. Helmy, and C. J. Kuo, “TCP over multi-hop 802.11 networks: Issues and performance enhancement,” in Proc. ACM Mobihoc, 2005,pp. 277–287.

[20] E. Altman and T. Jimenez, “Novel delayed ACK techniques for improving TCP performance in multi-hop wireless networks,” in Proc. Pers. Wireless Communication., 2003, pp. 237–253.

[21] NS-2 with Wireless and Mobility Extensions. [Online]. Available:http://www.monarch.cs.cmu.edu

[22] P. Gevros, J. Crowcoft, P. Kirstein, and S. Bhatti, “Congestion control mechanisms and the best effort service model,” IEEE Network, vol. 15, no. 3, pp. 16–26, May/Jun. 2001

[23] L. S. Brakmo and L. L. Peterson, “TCP Vegas: end to end congestion avoidance on a global internet,” IEEE J. Select. Areas Communication” vol. 13, no. 8, Oct. 1995, pp. 1465–1480.

[
24] P. Raptis1, V. Vitsas, K. Paparrizos, P. Chatzimisios, "Packet Delay Modeling of IEEE 802.11 Wireless LANs" Wireless Communications and Networking Conference WCNC. 2004 Vol.1 , 2004, pp: 213 – 218.

[25] Xin Ming Zhang, Wen Bo Zhu, Na Na Li, and Dan Keun Sung "TCP Congestion Window Adaptation Through Contention Detection in Ad Hoc Networks "IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, VOL. 59, NO. 9, NOVEMBER 2010, pp 4578- 4588

[26] K Chen, Y Xue, SH Shah and K Nahrstedt , "Understanding bandwidth-delay product in mobile ad hoc networks" Computer Communications Volume 27, Issue 10, 20 June 2004, Pages 923–934.

[27] K. Vijayalakshmi, S. Rajesh, R. Jayaparvathy, M. Sethuraman, S. Srikanth "ADAPTIVE TCP FOR 802.11 BASED ADHOC NETWORKS" National Conference on Communications (NCC), Indian Institute of Science, Bangalore, pp:494-498.

Appendix: Equations and Figures.

$$E_{ACK} = \frac{E_{ACK-1} \times (ACK - 1) + CRTT_{new}}{ACK} \quad \text{for } ACK < Nmax \tag{4}$$

$$V_{ACK} = \frac{(V_{ACK-1} + E^2_{ACK-1}) \times (ACK - 1) + CRTT^2_{new}}{ACK} - E^2_{ACK} \quad \text{for } ACK < Nmax \tag{5}$$

$$E_{ACK} = E_{ACK-1} \frac{CRTT_{new} - CRTT_{old}}{Nmax} \quad \text{for } ACK >= Nmax \tag{6}$$

$$V_{ACK} = \frac{(V_{ACK-1} + E^2_{ACK-1}) \times n + CRTT^2_{new} - CRTT^2_{old}}{Nmax} - E^2_{ACK} \quad \text{for } ACK >= Nmax \tag{7}$$

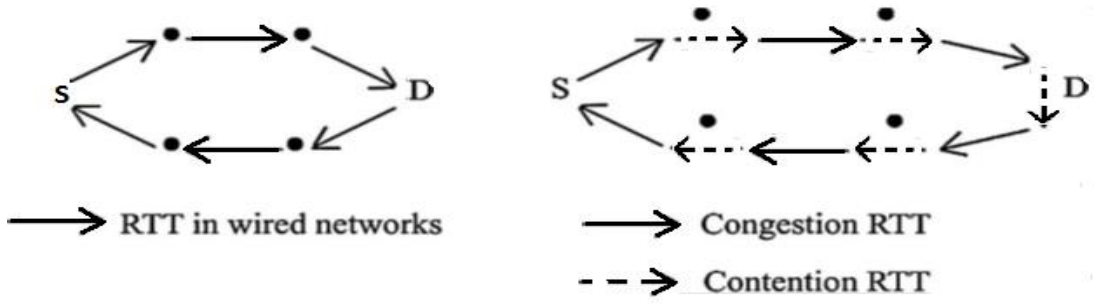

Fig. 2. Congestion RTT and contention RTT

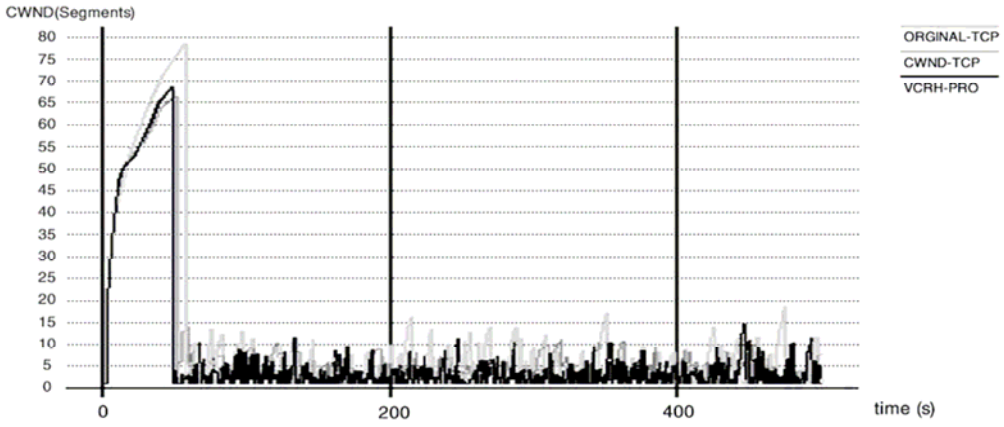

Fig 10 : CWND performance Graph of various algorithms in Wireless Network