ISSN NO. 2320-5407

Journal homepage: http://www.journalijar.com

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH

RESEARCH ARTICLE

AN EXTENDED STUDY ON NEWTON RAPHSON CONGESTION CONTROL.

Dr. J. VijiPriya[1], Dr. S. Suppiah[2].

1. School of Informatics, Hawassa University, Awassa, Ethiopia.
2. VelTech University, TamilNadu, India

Manuscript Info

Manuscript History:

Received: 14 December 2015
Final Accepted: 18 January 2016
Published Online: February 2016

Key words:
MANET, Newton Raphson, Bandwidth Utilization, Throughput and Fairness.

Corresponding Author

Dr. J. VijiPriya.

Abstract

The present Transmission Control Protocol runs on wired, wireless and heterogeneous networks. The original TCP is not so efficient for wireless and heterogeneous network environments where the packet losses are due to bit errors and handoffs. Although different procedures may attain diverse performance enlargements in several network situations, designing a congestion control algorithm that performs well across wired-cum-wireless network and Mobile Adhoc Network conditions remain a great challenge. Newton Raphson Congestion Control algorithm had been proposed and tested over wired-cum-wireless network. These experimental results were significantly improved as compared to other high speed TCP variants. In this comprehensive study, the performance of this proposed Newton Raphson Congestion Control TCP with high speed TCP variants over Mobile Adhoc Network has been proved and the investigational results were also improved considerably in the network evaluation constraints include bandwidth utilization, throughput and fairness in sharing bandwidth.

1. Introduction:-

The standard TCP on a wireless host is used to communicate with a fixed network. The innovative TCP is not consequently proficient in Mobile Adhoc Network wherever the packet losses are due to mobility, channel fading, limited resource and frequent network partition. . As a result, wireless Internet requires an optimized reliable transport protocol. The normal TCP assumes that the cause of any packet loss to be the network congestion. In this process, they reduce the congestion size to a minimum. The congestion window reduction minimizes the transmission rate and degrades the performance of TCP network. In addition, TCP invokes slow start causing long pauses of communication which may be unused even after the mobile recovery node from the temporary disconnection.

The present TCP variants are found to perform poorly over wireless and/or high Bandwidth Delay Product (BDP) links. To increase TCP performance over wireless networks and high BDP networks, many TCP variants have been recommended, including TCP Westwood, TCP Veno for wireless applications and Compound TCP, TCP CUBIC, FAST TCP for high BDP networks. Even though these procedures have succeeded in their corresponding objective applications, designing a TCP congestion control algorithm that performs gracefully in both wireless and high BDP networks is still a great challenge.

The high speed TCP variants include BIC-TCP, HS-TCP, H-TCP, Scalable-TCP, TCP-Reno, TCP-Vegas and TCP-Illinois. These variants reduce the packet loss rate by reducing their transmission rate. Also, these protocols are expected to retransmit lost packet and retransmission behavior was too aggressive that results in congestion over wireless networks. To overcome this problem, the Newton Raphson Congestion Control TCP called NRC-TCP had been proposed and tested in heterogeneous network. The NRC-TCP performed better than the other existing TCP variants. The present research work of the NRC-TCP with the existing TCP variants has been extended and verified

in Mobile Adhoc Network. The proposed Newton Raphson Congestion Control algorithm can also bring better performance better than the current high speed TCP variants.

The rest of the study arranged as follows, Section 2 gives the overview of existing TCP variants. Section 3 describes the property of NRC-TCP in detail, Sections 4 and 5 presents the results of emulation-based experiment and performance measured over Mobile Adhoc Network and Section 5 gives conclusion and future works.

2. Background and Motivations:-

The congestion control procedure of the generally used transport protocol TCP is accountable for discovering and responding to overloads in the Internet and has been the vital to the Internet's operative success. On the other hand, as the link capability grows and novel Internet solicitations with high-bandwidth demand emerge, the performance of the TCP develops inadequate, specifically on high speed and long distance networks. The foremost goal is the conventional behaviour of TCP in regulating its congestion window that guides the TCP sender transmission rates. To devastate these problems of TCP by adjusting its congestion window, a number of solutions have been proposed: BIC-TCP, CUBIC-TCP, TCP-Reno, TCP-NewReno, TCP-Compound, TCP-FAST, HS-TCP, H-TCP, S-TCP, TCP-Westwood, TCP-Africa and Predictive Congestion Control Protocol. These new protocols promise to improve TCP performance on high-speed networks significantly and are usually called TCPs for high-speed networks (Dave et al., 2013; Cheng et al., 2004; Ren et al., 2005; Rad and Kourdy, 2012; Subburam and Khader, 2012; Tiyyagura et al., 2011).

End-to-End Congestion Control algorithms propose an attractive approach to Internet congestion control, both in simplicity and scalability. End-to-End algorithms can infer the state of the network more accurately and make better decisions. The packet losses and latency (Delay) variations are signals of network congestion to an end-to-end algorithm.

Thus, the researchers have focused on three types of algorithms namely:

1. **Loss-based algorithms:-** use packet losses alone to react to network congestion
2. **Delay-based algorithms:-** use delay measurements alone to infer router queue occupancy and act before heavy congestion occurs
3. **Hybrid-based algorithms:-** use techniques from both loss-based and delay-based algorithms.

Here, few of the most relevant works on TCP end-to-end congestion control are presented.

2.1 Loss Based Congestion Control:-

Generally, TCP has used packet losses to detect network congestion. Many new algorithms now use delay information, but pure loss-based algorithms are still dominant in the Internet. Even if a number of researches have been conducted and protocol modifications recommended, the motivation behindhand the investigation of TCP variants is that each type has a number of extraordinary conditions, such as the outdated TCP has become known as TCP-Tahoe. TCP-Reno enhances unique novel Fast Recovery mechanism to TCP-Tahoe (Mathis et al., 1996). TCP-NewReno uses the latest retransmission mechanism of TCP-Reno. TCP-SACK authorities the receiver to specify several supplementary data packets that have been received out-of-order within one DUPACK (Fall and Floyd, 1998). TCP-FACK is TCP- Reno with forward acknowledgement (Renaud, 1988).

TCP-Tahoe:-

TCP-Tahoe engages three congestion control algorithms namely: Slow Start, Congestion Avoidance and Fast Retransmission. Jacobson and Karels (1988) established the fundamental algorithms for congestion avoidance and control. TCP-Tahoe retransmits the lost packet upon receiving three duplicate ACKs without waiting for retransmit timer to expire. It then sets the ssthresh to cwnd/2 and the cwnd to 1 MSS and enters into the Slow Start phase.

Later, it was found that this algorithm works well for a single packet drop but fails in case of multiple packet drops within a window of data. Each retransmission of packet will force TCP Tahoe to enter Slow Start phase thus resulting in serious loss of performance. This problem of TCP Tahoe is moderately enriched in TCP Reno by introducing Fast Recovery algorithm.

2.2 Delay Based Congestion Control:-

Delay Based Congestion Control algorithms are based around the premise that there are signs before network congestion occurs. Thus, they use network delay information, usually in the form of RTTs, to infer the state of queues at the routers. The delay-based approach is networking friendly than the loss-based methodology. It can avoid the typical congestion window oscillation of loss-based algorithms. In Pure delay-based algorithms, there are a number of applications for congestion control algorithms based on delay information. Wang and Crowcroft (1992) studied that RTT deviations are used to adjust the congestion window size. The congestion window grows according to Reno. But, if the value of the current measured RTT is higher than the average of the minimum and maximum measured RTTs so far, then the window size is decreased.

TCP-FACK:-

TCP Forward Acknowledgement (FACK) decouples congestion control from data recovery thereby attaining more precise control over the data flow in the network. The purpose idea of FACK algorithm is to consider the most forward selective ACK sequence number as a sign that all the previous unacknowledged segments were lost. This observation allows improving recovery of losses significantly. TCP-FACK provides congestion avoidance and fast retransmission mechanism, it faces so many situations for recovery and it is not possible to implement easily.

2.3 Hybrid Based Congestion Control:-

Hybrid-based TCP congestion control algorithms promises increased link efficiency, while retaining some of the fairness and network friendliness properties of pure delay-based proposals. Moreover, their loss-based component allows hybrid-based algorithms to remain aggressive enough while competing for bandwidth with pure loss-based algorithms.

Compound TCP:-

Tan et al. (2006) proposed Compound TCP to improve link efficiency and RTT fairness. It maintains two auxiliary congestion control windows. A traditional loss-based congestion window that follows Reno AIMD behaviour and a scalable delay-based window inspired in Vegas (Brakmo and Peterson, 1995). The resulting congestion window is the sum of the loss-based and the delay-based windows. If the network link is under-utilized, the delay-based component will quickly increase the congestion window to use available bandwidth, using behaviour similar to High Speed TCP (Floyd, 2003). When congestion starts to build up, the delay-based component will reduce the window. Under heavy network congestion, Compound TCP reverts to Reno behaviour.

3. NRC-TCP:-

Newton Raphson Congestion Control TCP uses the value of the prior congestion window value to compute its current congestion window value. It behaves like traditional TCP when the congestion window is below a threshold value. Threshold value is initialized by the receiver window. Above the threshold, NRC-TCP acts more hostilely in achieving bandwidth by growing its congestion window size aggressively. It generalizes Additive Increase and Multiplicative Decrease (AIMD). If the sending rate is too large between two communication hosts, it results in congestion. The router discards packets to evade congestion. As the sender detects to packet loss, it infers the occurrence of congestion in the network. NRC-sender will start a succession of congestion control at this moment and reduces the sending rate. Thus the following three algorithms are implemented in the proposed NRC-TCP method (VijiPriya and Suppiah, 2013). These algorithms are presented together with their pseudo codes:

1. NRC Slow Start
2. NRC Congestion Avoidance
3. NRC Retransmission

3.1 NRC Slow Start:-

NRC-TCP contrasts from other procedures during its slow start phase. The reason for this modification is that when a connection started first, it had no idea of the available bandwidth. It is possible that the bandwidth is increased exponentially by an appreciable amount only every other Round Trip Time. In this way the actual sending throughput to the expected is computed. When the congestion window exceeds or equals the slow start threshold, it exits NRC slow start and enters the NRC congestion avoidance phase. The Pseudo code of NRC Slow Start is given below:

NRC Slow Start:-

Initialization:-
Congestion Window ← 2* Maximum Transmission Unit (MTU);
Slow Start threshold ← Receiver Window;
IF (ACK is received or every Round Trip Time):
IF (Congestion Window < Slow Start Threshold):
Increment Congestion Window by 1 Maximum Segment Size (MSS);
ELSE
Perform NRC Congestion Avoidance;

3.2 NRC Congestion Avoidance:-

When the congestion window exceeds or equals slow start threshold, the state enters NRC congestion avoidance. The congestion window (w) is increased by e^{α} for every single arrival of a new acknowledgement until congestion occurs. Here α is a window scaling factor determined by real root of algebraic equation which is found by implementing Newton Raphson method. The Pseudo code is given below:

NRC Congestion Avoidance:-

/* NRC Slow Start is over */
IF (ACK is received or every Round Trip Time):
IF (Congestion Window ≥ Slow Start Threshold):
NRC Congestion Avoidance;
W' = w + e^{α}; if no loss;
ELSE
Perform NRC Slow Start;

3.3 NRC Retransmission:-

As packet loss happens, the TCP sender receives three duplicate ACK and triggers modified fast retransmission and fast recovery immediately. Subsequently, the sender does not wait for retransmission timeout to send lost packet back. Besides slow start threshold value will be set as e^{β} or double the Maximum Transmission Unit and set up the congestion window as slow start threshold value plus 3 MTU, where β is a window scaling factor determined by real root of algebraic equation which is found by implementing Newton Raphson method. In fast recovery procedure the sender has to wait until a retransmission timeout occurs or an ACK acknowledges all of the data including the data that was outstanding after its commencement. The Pseudo code is as follows:

NRC Fast Retransmission:-

/*Modified Slow Start and Modified Congestion Avoidance*/
IF (3 Duplicate ACKs are received or Retransmission Time is expired):
Modified Fast Retransmission and Fast Recovery;
W' = w – e^{β}; if loss occurs;
After retransmission, Enter Modified Fast Recovery:
Congestion Window = Slow Start threshold + 3 MSS;

3.4 Comparison of Window based TCP Congestion Control:-

The performance of all the pseudo codes implemented for Mobile Adhoc Network are presented in Table 3.1.From table it can be observed that the algorithm based on NRC-TCP performs better with regard to parameters such as bandwidth utilization, maximum throughput, and fairness. It is to be noted that these parameters are the main scope of this study.

Comparison of window based High Speed TCP Congestion Control is presented in Table 1. It is observed that NRC-TCP is able to increment congestion window rapidly. Thus, it can attain whole of available bandwidth by increasing the window accordingly.

Table 1 Comparison of Window based TCP Congestion Control

TCP Variants Modification Schemes	In case of receiving an ACK	In case of packet loss	Congestion Control Mechanism
HS –TCP	cwnd = cwnd + α / cwnd	cwnd = cwnd * (1–β)	AIMD
H- TCP	cwnd=cwnd+2 (1 – β) α/cwnd	cwnd = cwnd * β	AIMD
Scalable-TCP	cwnd=cwnd+ α	cwnd=cwnd* β	AIMD
BIC – TCP	cwnd = cwnd + 1/cwnd; or cwnd=cwnd+ (target_win-cwnd)/ cwnd; target_win= (max_win+min_win) /2;	cwnd= W_{max} * (1 – β)	Binary Search Increase and AIMD
CUBIC-TCP	cwnd= C (t- K) ^{3}+W_{max}	cwnd= β W_{max}	AIMD
Proposed NRC-TCP	**cwnd=cwnd + e^{α}**	**cwnd=cwnd – e^{β}**	**AIMD**

4. Simulation:-

The proposed work was tested with congestion imposed in the following network to study the performance of NRC-TCP by varying bandwidth and RTT of bottleneck link over Mobile Adhoc Network.

4.1 Mobile Adhoc Network:-

The MANET topology designated for the experiments is shown in Fig.1. It consists of 50 mobile devices which were connected by wireless duplex links. Each device has been moved independently in any direction using random motion. Each node was acting as both host and router. The nodes transmitted and received their own packets. The nodes which used its Ad-hoc routing protocol (DSDV) to route the packets to its target node. The traffic used File Transfer Protocol (FTP). Here, the maximum number of packets in Queue is 50. X and Y dimensions of the topography are 600 and 800 respectively.

The experiments on the above topology were carried out to test the performance of the new NRC congestion control mechanism with the previous high speed TCP variants such as BIC-TCP, CUBIC-TCP, High Speed TCP, Hamilton-TCP and Scalable TCP using Network Simulator (NS2).

The proposed NRC-TCP belongs to end hosts rather than intermediate routers in the network, in which the end points were responsible for performing the necessary changes to ensure a good adaptation and do not need any support from the intermediate nodes. The main advantage of this scheme is that they can be used in any situation.

5. Results and Discussion:-

Various experiments on the mobile Adhoc Network topology have been conducted to test the performance of proposed NRC-TCP with existing end-to-end high speed TCP variants such as BIC-TCP, CUBIC-TCP, HS-TCP, H-TCP, and Scalable-TCP based on the three TCP performance evaluation constraints namely, full bandwidth utilization, throughput, fairness in sharing bandwidth.

5.1 NRC-TCP Full Bandwidth Utilization:-

Many experiments have been performed with flows of NRC-TCP, BIC-TCP, CUBIC-TCP, HS-TCP, H-TCP, and Scalable TCP on MANET. The experiments were run for 600ms and the congestion window for each flow has been measured. From this Fig. 2, it can be observed that the NRC-TCP is capable to increase its congestion window very rapidly so that it can achieve the whole of available bandwidth by growing the congestion window accordingly.

5.2 NRC-TCP Throughput:-

Experiments regarding TCP-Throughput were conducted for 600ms. The average throughput for each flow has been measured. Fig. 3 shows the comparison of the accumulated throughput. From this figure, it can be seen that a better

performance with the NRC-TCP which gives an improvement in the value of mean throughput obtained. Also the value of mean throughput of the existing algorithms is low.

5.3 NRC-TCP Fairness:-

The experiments were run for 600ms with flows of NRC-TCP, BIC-TCP, CUBIC-TCP, HS-TCP, H-TCP, and Scalable TCP and the average throughput for each flow has been measured. Fig. 4 shows that High Speed NRC-TCP performs better in sharing the bandwidth compared to previous High Speed TCP Variants.

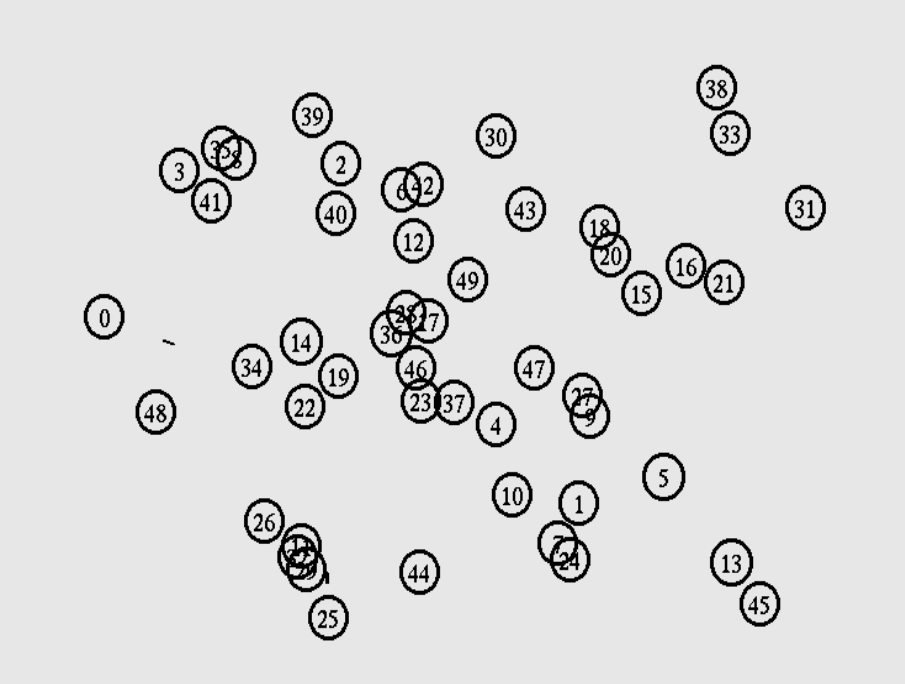

Fig.1 Mobile Adhoc Network Topology

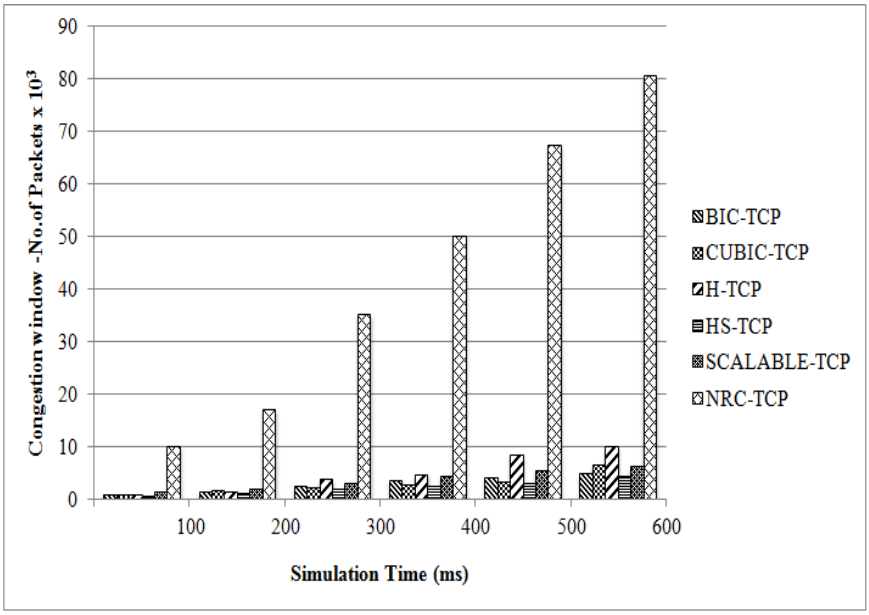

Fig.2 NRC-TCP Full Bandwidth Utilization

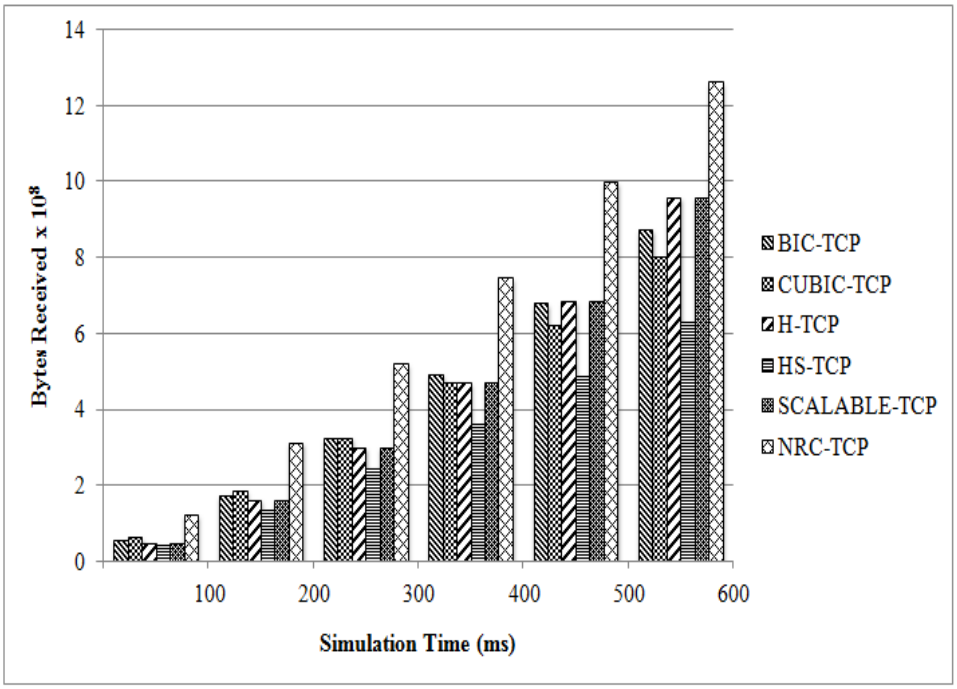

Fig. 3 NRC-TCP Accumulated Throughputs

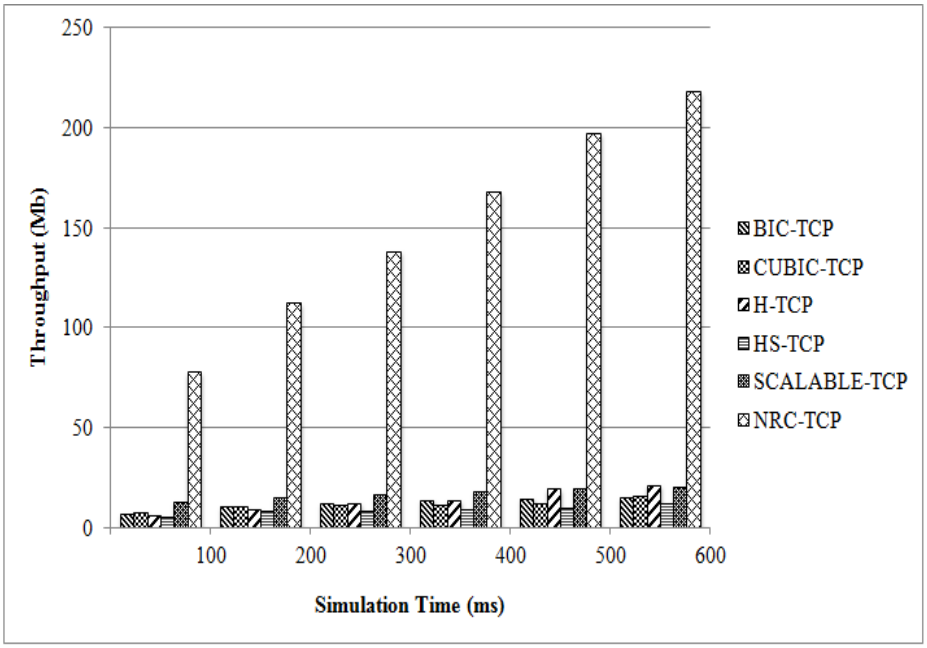

Fig. 4 NRC-TCP Fairness

6. Conclusions:-

All Internet applications are essentially based on Transmission Control Protocol due to its reliability. The existing high speed TCP variants such as HS-TCP, H-TCP, Scalable-TCP, BIC-TCP and CUBIC-TCP are less efficient due to non-utilization of full bandwidth and misinterpret wireless loss as congestion loss in wireless networks. To overcome these deficiencies of different schemes, the new proposed Newton Raphson Congestion Control TCP has

also been attempted in the extended present research work. The simulations with the Network Simulator 2 (NS-2) tool are to verify the performance of the proposed research work. The simulation results focus on network parameters such as Bandwidth Utilization, Throughput and Fairness. The outcome of the present research work has been compared with other high speed TCP Variants over MANET. The NRC TCP performs better in congestion window growth, mean throughput and sharing the bandwidth than the other protocol.

References:-

1. Dave, H., Gupta, V. & Dihulia, P. (2013): Performance comparison between TCP sack and TCP Vegas using NS-2 Simulator. Int. J. Comput. Applic., 68: 49-52.
2. Cheng Jin, David Wei, X. & Steven Low, H. (2004): FAST TCP: motivation, architecture, algorithms, performance. INFOCOM., pp. 157-165.
3. Ren Wang, Kenshin Yamada, M., Yahya Sanadidi, & Mario Gerla, (2005): TCP with sender-side intelligence to handle dynamic, large, leaky pipes. IEEE Journal on Selected Areas in Communications.,23(2): 235-248.
4. Rad, MRN & Kourdy, R. (2012): TCP NewReno buffer management in network on chip. J. Comput., 4: 2151-617.
5. Subburam, S., & Khader, PSA. (2012): Predictive congestion control mechanism for MANEY. Int, J. Comput, Sci. Eng., 3: 64-69.
6. Tiyyagura, S., Nutangi, R. & Reddy, PC. (2011): An improved snoop for TCP Reno and TCP sack in wired-cum-wireless networks. Int, J. Comput, Sci. Eng., (2)455-460.
7. Mathis, M., Mahdavi, J. & Floyd, S., Roma, A. (1996): RFC 2018: TCP Selective acknowledgment options.
8. Fall, K., & Floyd, S.(1998): Simulation Based Comparison of Tahoe, Reno and SACK TCP. IEEE/ACM Transactions on Networking., 22(8) 34-54
9. Renaud Bruyeron, Bruno Hemon, Lixia Zhang. (1998): Experimentations with TCP Selective Acknowledgment. ACM SIGCOMM Computer Communication Review., pp. 10-25.
10. Jacobson, V., & Karels, MJ. (1988). Congestion avoidance and control. In ACM Computer Communication Review; Proceedings of the Sigcomm‘88 Symposium., 18: 314–329.
11. Wang, Z., & Crowcroft, J. (1992): Eliminating Periodic Packet Losses in 4.3-Tahoe BSD TCP Congestion Control Algorithm. ACM Computer Communication Review. 22(2): 9–16.
12. Tan, K., Song, J., Zhang, Q., & Sridharan, M. (2006): A Compound TCP Approach for High speed and Long Distance Networks. In proc. IEEE INFOCOM, Barcelona, Spain, pp. 89-98.
13. Brakmo, LS., & Peterson, LL. (1995): TCP Vegas: end to end congestion avoidance on a global internet. IEEE Journal on Selected Areas in Communications., 13(8): 1465–1480.
14. Floyd, S. (2003): High-Speed TCP for Large Congestion Windows. IETF, RFC 3649.
15. VijiPriya, J., and Suppiah, S. (2013): Application of Newton Raphson Algorithm for Optimizing TCP Performance, Journal of Computer Science., 9(5):566-571