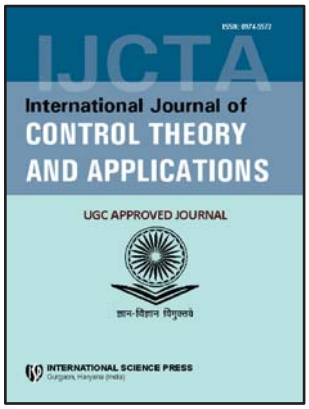

International Journal of Control Theory and Applications

ISSN : 0974-5572

Volume 10 • Number 31 • 2017

Controlling Congestion for MANETs using TCP – Vegas Based on Cuckoo Search

K. Praveen Kumar Rao[a] and T. Senthil Murugan[b]

[a]Research Scholar, CSE Department, Vel Tech Dr. RR & Dr. SR Technical University, Chennai, India

E-mail: praveenkumarrao.k@gmail.com

Associate Professor, CSE Department, Vel Tech Dr. RR & Dr. SR Technical University, Chennai, Indiah

E-mail: senthilmuruganme@gmail.com

Abstract : TCP Vegas is a source calculation that offers generally rich performance in the Internet congestion control. TCP Vegas has a few issues which have an impact on its performance. The major issue is related to Rerouting. Another critical issue arises when a flow joins to the network later than the previous flows; it has the problem of congested queues. It considers wrongly the measured round trip time as its initial $Base_{RTT}$. It implies that while other flows diminish their sending rates because of the existing congestion, this flow does not detect the congestion and subsequently increases its sending rate. These issues chiefly have roots in the Vegas estimation process of the propagation delay *i.e*. $Base_{RTT}$. In this paper, a novel algorithm is proposed which utilizes the cuckoo search optimization for selecting the ideal estimation of $Base_{RTT}$. In addition, our proposed calculation powerfully picks moderate begin calculation in light of the assessed continuously the accessible data transfer capacity and alter diminish / increment rate in blockage shirking stage as per particular system environment. Reenactment comes about demonstrates that our proposed calculation can viably stay away from parcels misfortunes and achieve the greatest throughput when contrasted and existing calculations.

Keywords : *Congestion Control, Cuckoo Search Optimization, Fairness, Re – routing, Slow Start, Packet loss, TCP – Vegas, Throughput.*

1. INTRODUCTION

With the increasing demand for connectivity, there is a necessity for mobile wireless communication. The use of laptops and various handheld devices is increasing rapidly in day – to – day life. All communication devices have the support of a fixed Base Station (BS) or Access Points (AP) that corresponds to the last – hop – wireless model. This can be observed in wide – area wireless cellular systems. Such a support is not available in settings where access to a wired infrastructure is not possible. This led to the development of Mobile Ad – hoc Networks (MANET's) [1] [2].

MANET is the most important network because of its use and fast deployment. MANETs are attracted as an intension for supporting mobile – wireless devices, with continuous connectivity regardless of their high mobility. Thus, MANETs are under intense focus in research, to improve their performance. A Mobile Ad hoc NETwork (MANET) is a dynamically changing network of mobile hosts (MHs) that communicate without support of any fixed infrastructure. There is always a direct communication among neighboring devices. But, the communication between non – neighboring nodes who act as routers, requires various routing algorithms such as Destination Sequenced Distance Vector (DSDV), Ad – hoc On – Demand Distance Vector (AODV), and Dynamic Source Routing (DSR) [3],[4]. Because of the nature of MANETs having characteristics like dynamic topology, high mobility of nodes, frequent route changes, network partitions, wide range of channel conditions, etc., designing new protocols or applying the existing protocols (proposed for fixed networks) is a challenge [5].

Transmission Control Protocol (TCP) is widely used in Internet. Therefore, using TCP to provide a reliable end – to – end message delivery in MANETs is necessary in order to achieve smooth integration with the wired Internet. TCP is a reliable connection oriented end – to – end protocol. TCP contains within itself the mechanisms for ensuring reliability by requiring the receiver acknowledge. The network in a MANET is not perfect and small percentages of packets are lost in the route, either due to network error or due to congestion in the network. The loss incurred due to network congestion is minimal and most of the packet losses are due to buffer overflows at the router. Hence, it is important for TCP to react to a packet loss and take action to reduce congestion. TCP tries to ensure reliability by always starting a timer whenever it sends a packet or segment. If sender does not receive an acknowledgement from the receiver within the specific time – out interval, then it retransmits the packet or segment [6]. There are different congestion control techniques for various TCP variants. One of the best known TCP variant is TCP – Vegas. TCP – Vegas takes into account the existing network conditions. Some TCP variants make use of packet loss as an indication of network congestion. However, TCP – Vegas does not use packet loss as an indication of congestion, instead, it can detect the presence of initial network congestion earlier than any other TCP variants by analyzing the transmission rate in accordance to the variation in the estimated Round – Trip – Time (RTTs). TCP – Vegas increases / decreases its sending window size dynamically, according to the observed RTTs of sending packets, whereas other TCP variants only continue increasing its window size until packet loss is detected [5][7].

2. RELATED WORK

E. Abolfazli and V. Shah – Mansouri [8] developed a controller to adjust the queue levels at the intermediate routers when the sources employ TCP Vegas. Their proposed scheme was based on the integral back stepping technique while the controller signal is the TCP Vegas parameter. The controller is designed such that it is robust to changes in the network setting. This is validated through simulations.

Dongkyun Kim et al [9] focuses on addressing the inaccuracy problem of the $Base_{RTT}$ estimation, which is most critical for TCP – Vegas over MANETs because a route change (RC) can make the $Base_{RTT}$ used over a previous path obsolete. Hence, it was proposed that their TCP – Vegas – adhoc protocol, which was made aware of RCs uses the correct $Base_{RTT}$ values. Through simulations using NS – 2, it was observed that the TCP – Vegas – ad hoc outperforms the standard TCP – Vegas protocol, especially under high mobility scenarios over both reactive and proactive adhoc routing protocols.

Cheng-Yuan Hoet al [10] investigated how to improve the performance of delay-based TCPs with rerouting and propose a mechanism for delay – based TCPs. The proposed mechanism is able to re – measure the $Base_{RTT}$, if necessary by detecting the change in Time – To – Live (TTL) value of two end – hosts. Based on the simulation results, the proposed mechanism may improve the TCP performance with rerouting and in Mobile IP (MIP) networks and mobile ad hoc networks (MANETs).

Shubhada P. Deshmukh and Sanjesh S. Pawale [11] presented an algorithm, TCP Vegas – RB is to use the rerouting delay for estimation of RTT which improved the performance by detecting the packet loss. Delay – based TCP detect network congestion in the early stage and successfully prevent periodic packet loss that usually occurs in loss – based mechanisms. Their work focuses on how to improve the performance of delay – based TCPs with rerouting and proposes a mechanism for delay – based TCP Vegas to detect congestion by measuring RTT and to minimize the packet loss.

Shahram Jamali et al [12] proposed an algorithm named Pegas to improve the estimation process of $Base_{RTT}$ and to overcome rerouting and unfairness problems in TCP Vegas algorithm. Pegas formulated estimation of $Base_{RTT}$ as an optimization problem to solve by particle swarm optimization algorithm and sets it dynamically by considering network conditions. Pegas was easily applicable to the end hosts and estimates $Base_{RTT}$ very close to its true value. The estimation of $Base_{RTT}$, causes an increase in throughput and decrease in packet loss ratio.

3. TCP VEGAS

TCP – Vegas proposed in [5] also well known as TCP variant, takes into account the network traffic conditions. The congestion avoidance mechanism that TCP Vegas uses is quite different from other TCP variants. Other TCP variants use loss of packets as a signal to indicate there is congestion in the network and have no way of detecting any incipient congestion before packet losses occur. Thus, other TCP variant reacts to congestion rather than attempt to prevent the congestion. On the other hand, TCP Vegas uses the difference between the estimated throughput and the measured throughput as a way of estimating the available bandwidth (congestion state) in the network, regulate the transmission rate, and avoid congestion. The concept is that when the network is not congested, the expected throughput and the actual throughput are close together. Otherwise, the actual throughput will be much smaller than the expected.

$$\begin{cases} cwnd + 1 & diff < \alpha \\ cwnd - 1 & diff < \beta \\ cwnd & \text{otherwise} \end{cases}$$

$$diff = \text{Expected rate} - \text{Actual rate}$$

$$\text{Expected rate} = \frac{cwnd(t)}{Base_{RTT}}$$

$$\text{Actual rate} = \frac{cwnd(t)}{RTT}$$

where, $Base_{RTT}$ is the minimum encountered RTT of the connection, $cwnd(t)$ is the current congestion window size, RTT is the actual Round Trip Time, and α and β are the parameters whose values are set to 1 and 3, respectively. As a consequence, TCP – Vegas is capable of detecting network congestion in the early stages and prevents periodic packet losses.

4. PROBLEMS WITH TCP VEGAS

4.1. Re – routing

Re – routing is the major problem. When the route is changed by the network and Round Trip Time increase, Vegas mistakenly perceives the increased Round Trip Time as the result of network congestion and therefore decreases its own sending rate. This problem is because Vegas adjusts its own congestion window size *cwnd* based on $Base_{RTT}$.

4.2. Unfairness

When various sources consider different values for their $Base_{RTT}$, each node senses a different congestion level and therefore will have a different sending rate unfairly.

4.3. Slow – start

The slow – start algorithm always begins with sending one segment. It may take many RTTs to reach the optimal operating point, therefore resulting in poor utilization of the available bandwidth for short transfers.

4.4. Congestion control

In this phase, the original TCP Vegas algorithm adjusts the value of *cwnd* by a constant number, without having any knowledge of how much the ratio of expected transmission speed is with respect to the current transmission speed.

5. PROPOSED APPROACH

The novel approach tries to solve the problems existing in TCP – Vegas, by introducing a different methodology. The main issue with TCP – Vegas is Re – routing and unfairness. These are achieved because of the inaccurate estimation of $Base_{RTT}$. The accurate estimation of $Base_{RTT}$ can solve the re – routing and unfairness problem and can lead to an improvement in the performance of the network. For obtaining the accurate information of $Base_{RTT}$, Cuckoo Search optimization algorithm is used. The Cuckoo Search algorithm searches the solution space to find the accurate value for $Base_{RTT}$. In the proposed methodology, it dynamically adjusts the value of ssthresh with respect to the default setting based on the available bandwidth. Based on updated ssthreshvalue, the mid – speed start methodology is applied to increase the utilization of the network. In the proposed algorithm, *cwnd* is dynamically changed according to current congestion degree.

5.1. Cuckoo Search Algorithm

Cuckoo Search is a meta – heuristics algorithm. It is based on the behaviour such as brood parasitism of certain species of cuckoos. The basic idea applied is the aggressive reproduction strategy of cuckoo and the usage of Levy flights. It is widely used in engineering optimization problems with exceptionally good results.

The following is the technique used.

1. Each cuckoo lays one egg at a time, and dumps its egg in a randomly chosen nest.
2. The nests with high quality of eggs, will carry over to the next generations.
3. The number of host nests available is fixed. The egg laid by a cuckoo is discovered by the host bird with a probability $\in$pa[0, 1]. The last assumption can be approximated by the fraction pa of the n nests that are replaced by new nests with random solutions.
4. Here, each egg in a nest represents a solution.
5. Each cuckoo egg represents a new solution.
6. The aim is to use the new and potentially better solutions to replace a not – so – good solution in the nests.

When generating new updated solutions $x^{(t+1)}$, the levy flight for the i^{th} solution is represented by the following equation

$$x_i^{(t+1)} = x_i^t + \alpha \otimes \text{Levy}(\lambda)$$

where $x_i^{(t+1)}$ indicates the updated solution, x_i^t is the position of the current solution where t is the iteration number, and $\propto$ represents the step size that should be related to the scales of the problem of interests, where $\propto \geq 0$. The product $\otimes$ means entry – wise multiplications and Levy(λ) is the probability distribution, where, $1 \geq \lambda \geq 3$.

5.2. Dynamic TCP – Vegas

In this work, TCP – Vegas is enhanced with adaptively allocating the values of $Base_{RTT}$,ssthresh, and ratio of increasing or decreasing the size of congestion window (*cwnd*). The $Base_{RTT}$ values are optimally obtained from the Cuckoo Search optimization algorithm. Based on the available bandwidth, the values of ssthresh and ratio of increasing the size of *cwnd* are calculated.

5.2.1. Optimization of $Base_{RTT}$

The TCP Vegas performance strictly depends on accurate estimation of $Base_{RTT}$ as explained in the above sections. As a consequence, any carelessness in the estimation of $Base_{RTT}$ can lead to the degradation of performance in the network. If $Base_{RTT}$ is underestimated, throughput will be very low. Also, if $Base_{RTT}$ is overestimated, packet loss rate will be increased. PSO optimization approach for accurate, dynamic and evolutionary estimation of $Base_{RTT}$ is used to address the problem.

In this scheme, considering the current value of $Base_{RTT}$ as the particle position, it is attempted to direct it to the optimal point. From the view of congestion avoidance, an optimum trajectory in the evolutionary estimation of $Base_{RTT}$, is one in which the throughput and utilization has increasing trend and at the same time packet loss ratio is decreasing across iterations.

$$max(\text{Base}_{\text{RTT}}(k)) = \frac{\text{Actual Rate Avg}(k)}{\text{Actual Rate Avg}(k-1)} \times \frac{c(k-1) - \text{Drops}(k)}{c(k-1)}$$

where, k is the iteration number of the optimization computation, Actual Rate Avg(k) is the average actual rate of the sending source at iteration k, Drops(k) is the number of packets dropped at iteration k, and $c(k-1)$ is the total number of packets sent during iteration $(k-1)$. Here, iteration refers to an interval during which the required parameters are measured and lasts around one RTT. Hence, the available delay in the network is involved in this formula.

In the above objective function, Actual Rate Avg(k)/ Actual Rate Avg($k-1$) aims in increasing the sending rate and $((c(k-1) - \text{Drops}(k))/ c(k-1))$ intends in decreasing the dropped packets ratio. The objective function can only be maximized by the true value of $Base_{RTT}$, because it is the only point at which throughput is high and drop rate is low. The objective function is also affected by other factors which include sending rates of other active flows. Hence, it is very difficult and even impossible to achieve an accurate mathematical relation among the objective function and corresponding $Base_{RTT}$.

5.2.2. Dynamic slow start and congestion avoidance

TCP Vegas is one which adjusts the sending rate to keep network transmission steady. The weakness is with modifying the value of *cwnd* by a constant number, such that the speed is changed slowly. Hence, the idea is to modify *cwnd* dynamically by delta and alpha. The theme of modification is done by dynamically choosing a strategy that fits the bandwidth. In slow start phase, the idea is to dynamically choose slow start algorithm. In congestion avoidance phase, the congestion window is adjusted by the ratio of difference in predictive size and real size, to improve TCP Vegas efficiency.

In the first step, detecting the bandwidth of transmission route is done by the following equation.

$$\Delta = (\text{Expected rate} = \text{Actual rate}) \times \text{Base}_{\text{RTT}}$$

Once the bandwidth is calculated, subsequently the proposed algorithm calculates the value of ssthresh dynamically based on the estimated bandwidth instead of the default value. The expression of ssthresh is equated as

$$\text{ssthresh} = \text{bandwidth}(\Delta) \times \text{RTT}$$

The value of ssthresh is adjusted based on the estimated bandwidth. At the beginning of the slow – start phase, the sender estimates the available bandwidth, then updates the value of ssthresh. In the process of slow – start, the proposed algorithm adjusts in real time the value of ssthresh to adapt to the current link state by setting the initial size of *cwnd* with half of ssthresh which is called as mid – speed start. On the other hand, the proposed algorithm improves the increment dynamically by adjusting the size of *cwnd* instead of the slow – start phase. The traditional TCP increases exponentially the size of *cwnd* from one to the value of ssthresh.

This fixed increment is easy to cause the low bandwidth utilization rate during the initial stage of slow start. In addition, when the size of *cwnd* soars to half of the value of ssthresh, the way of exponential increment is easy to cause congestion due to big amplitude change, leading to the oscillation of *cwnd* , triggering the resumption of slow – start, and seriously impacting on throughput. The weakness of this strategy is modifying the value of *cwnd* by a constant number so that the speed is changed slowly. Thus, a more adaptive idea is to modify *cwnd* dynamically by delta and alpha.

$$cwnd = \begin{cases} cwnd + \left(\dfrac{\alpha - \Delta}{\alpha}\right) & \Delta < \alpha \\ cwnd - \left(\dfrac{\Delta - \beta}{\beta}\right) & \Delta < \beta \\ cwnd & \text{otherwise} \end{cases}$$

If delta is greater than beta, it means network bandwidth is congested, the greater the value of delta, the quicker the ratio is decreasing.

If delta is less than alpha, it indicates that network bandwidth is not utilized, and the sending rate should be increased.

The closer delta is to zero, the higher is the ratio.

However, in the congestion control phase, the increasing speed could not be too high. If delta is greater than alpha and less than beta, it reflects that the actual sending rate is close to current sending rate, and thus *cwnd* remains unchanged.

6. SIMULATION

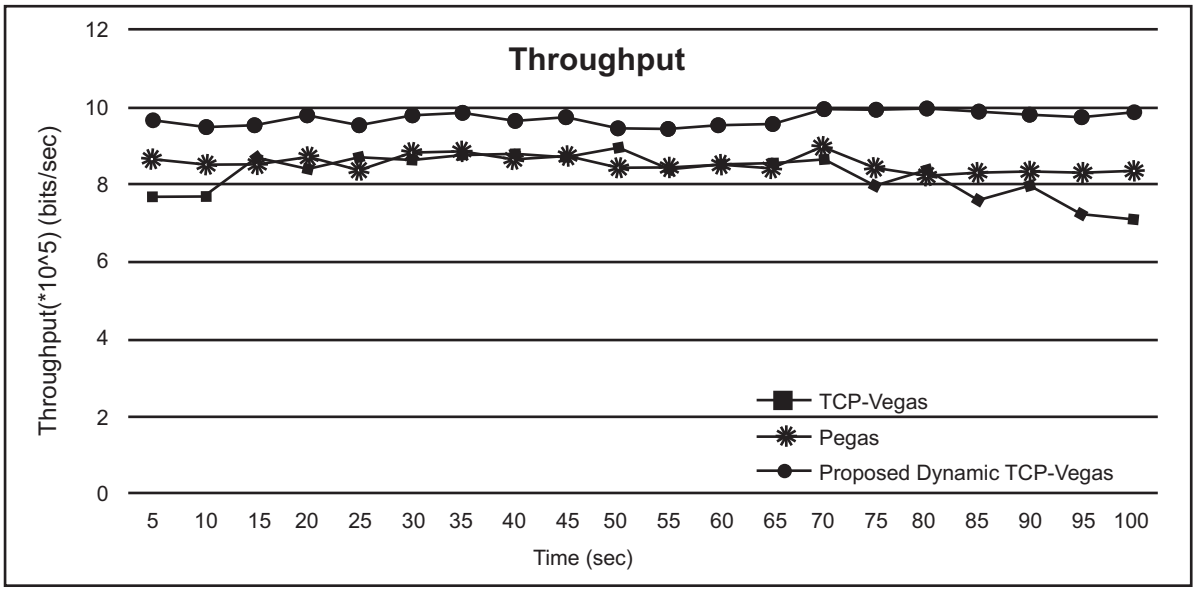

Figure 1: Comparative analysis of throughput

This section describes two simulations by using NS – 2. In the proposed work, we evaluate our methodology by comparing with the existing methodology such as Pegas [12] and traditional TCP – Vegas in terms of throughput and packet loss.

Figure 1 indicates the comparative analysis of throughput for various methodologies. By analyzing the above figure, the traditional TCP – Vegas has little throughput when compared with the Pegas and the proposed methodology. Pegas performs well than the traditional TCP – Vegas in terms of throughput. The reason behind this is, it optimally selected the $Base_{RTT}$ value by the

PSO algorithm, but their methodology takes more time to get maximum throughput because *cwnd* increase one by one. The proposed methodology dynamically updates the size of congestion window based on the value of delta, alpha and beta which leads to obtain the maximum throughput in a minimum duration of time.

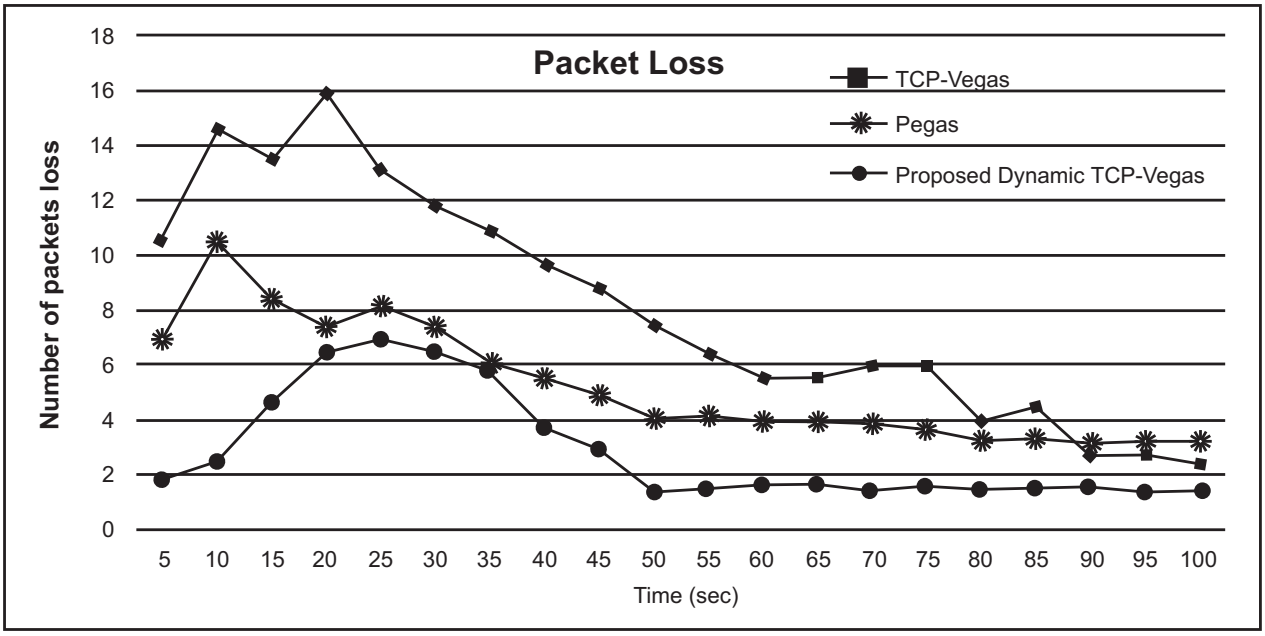

Figure 2 : Comparative analysis of packet loss

Figure 2 indicates the comparative analysis of packet loss for various methodologies. By analyzing the above Figure 1, the traditional TCP – Vegas has more packet losses when compared with Pegas than the proposed methodology. Pegas performs well than the traditional TCP – Vegas in terms of packet loss and the reason behind this, is it optimally selects the $Base_{RTT}$ value by the PSO algorithm. However, the proposed methodology dynamically updates the size of ssthresh based on the available bandwidth which makes the proposed algorithm to reduce the number of packet losses and also update the size of congestion window based on the value of delta, alpha and beta which leads to obtain the minimum packet losses in a minimum duration of time.

7. CONCLUSION

This paper proposes an algorithm which uses the cuckoo search optimization algorithm for selecting the optimal value of $Base_{RTT}$ which leads to solve the re – routing problem and unfairness problem in the TCP – Vegas. The proposed algorithm also dynamically chooses slow start algorithm based on the estimated in real time available bandwidth and adjusts decreases / increases rate in congestion avoidance phase according to specific network environment. The simulation results show that the proposed algorithm can effectively avoid packets losses and attain maximum throughput when compared with the existing algorithms.

REFERENCES

[1] Royer, E.M., Toh, C.-K.: A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks. IEEE Personal Communications Magazine, 46–55 (1999)

[2] RaziaNisarNoorani, Asma Ansari, Kamran Khowaja, Sham-ul-ArfeenLaghari, and Asadullah Shah, "Analysis of MANET Routing Protocols under TCP Vegas with Mobility Consideration", Wireless Networks, Information Processing and Systems, Vol. 20 of the series Communications in Computer and Information Science, pp. 227-234, 2009.

[3] A. Ahuja et al., "Performance of TCP over different routing protocols in mobile ad hoc networks," in Proc. IEEE VTC, 2000, pp. 2315–2319

[4] Pratap K. Meher, P. J. Kulkarni, "Analysis and Comparison of Performance of TCP-Vegas in MANET", International Conference on Communication Systems and Network Technologies (CSNT), pp. 67 – 70, Katra, Jammu, 2011.

[5] Raed T. Al-Zubi, Marwan Krunz, Ghazi Al-Sukkar, Mohammed Hawa, Khalid A. Darabkh, "Packet Recycling and Delayed ACK for Improving the Performance of TCP over MANETs", Wireless Personal Communications, Vol. 75, No. 2, pp. 943-963, 2014.

[6] A.A.Hanbali, E. Altman, and P. Nain, "A Survey of TCP over Ad Hoc Networks," IEEE Communications Surveys & Tutorials, Third Quarter, pp. 22 - 36, 2005.

[7] Jonas Katlsson, Alba Batlle and andress J. Kassler, "TCP performance in Mobile Ad Hoc Networks connected to the internet", vol 10, Issue 1, 2007.

[8] E. Abolfazli and V. Shah-Mansouri, "Dynamic adjustment of queue levels in TCP Vegas-based networks", Electronics Letters, vol. 52, no. 5, 361 – 363, 2016.

[9] Dongkyun Kim, HanseokBae, and C. K. Toh, "Improving TCP-Vegas Performance Over MANET Routing Protocols", IEEE Transactions on Vehicular Technology, vol. 56, no. 1, pp. 372 – 377, 2007.

[10] Cheng-Yuan Ho, Yaw-Chung Chen, Cheng-Yun Ho, "Improving Performance of Delay-Based TCPs with Rerouting", IEEE Communications Letters, vol. 11, no. 1, pp. 88-90, 2007

[11] Shubhada P. Deshmukh and Sanjesh S. Pawale, "TCP Vegas rerouting detection and improving Performance", International Journal of Wired and Wireless Communications Vol.1, No. 1, 2012.

[12] ShahramJamali, NedaAlipasandi, BitaAlipasandi, "TCP Pegas: A PSO-based improvement over TCP Vegas", Applied Soft Computing, vol. 32, pp. 164–174, 2015.