IOSR Journal of Engineering (IOSRJEN)
e-ISSN: 2250-3021, p-ISSN: 2278-8719, www.iosrjen.org
Volume 2, Issue 10 (October 2012), PP 28-34

Memorized Window Consolidation (MWC) based TCP Recovery for Vegas and B/w Based Congestion Control

[1]Shree Ammu, [2]Prof Suresh Ballala
[1,2]Department of ECE, Sri Indu College of Engg and Tech.Affiliated to JNTU Hyderabad AP, India.

Abstract: *This paper presents a framework for Unanimous TCP Flow Window Size Recovery, "Memorized Flow Recovery", which differs from most TCP algorithms by Memorized Window Size Consolidation as measure to control the window size increment post link recovery from disconnection. It identifies the window size in use during certain iterations, and in event of link recovery from different conditions, system will uses most stabilized Percentile value to ensure Quick Recovery of Connectivity and Better Through put. Then, the stability of the mechanisms is also investigated via the simulations, and results of the proposed procedure is shown to better than Other Recovery mechanizes Currently being used i.e TCP-Vegas and Upcoming Congestion Control of TCP based on Bandwidth Estimation.*
Proposed Procedure is shown to be Easily Implemented with TCP-Vegas and TCP-based on B/w Estimation, and can be easily adapted in current other congestion control mechanisms and that may come in future.

I. Introduction

The Internet has become an un distinguishable Part of our Life Two main protocols are being implemented in the transport layer of the Internet, namely, UDP (User Datagram Protocol) and TCP (Transmission Control Protocol). They are distinguished by their connection type. UDP is a connectionless protocol that suits for multimedia transmissions; on the other hand, TCP is a connection-oriented Protocol used to provide a reliable transmission policy in an unreliable network, which may vary due to different users, routers, bandwidths, LAN, WAN, MAN or Mobile IP.

Operation of TCP in Un-Reliable Network make it highly Vulnerable to various Forms of Losses and Disconnection. In recent studies, Low *et al*. pointed out that 90% of the Internet traffic is TCP-based, which is the main workhorse and causes congestion (Low *et al*., 2002), which can with additional vulnerability of Losses and disconnections. To avoid this problem, controlling the sending rate is necessary and the issue has been discussed and developed over the past quarter of a century.

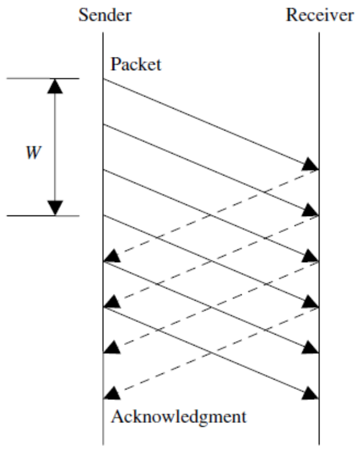


Fig. 1 Illustrative example of window size

In 1988, a major breakthrough was achieved by Jacobsen, who proposed a mechanism called Tahoe, which made the congestion window change dynamically (Jacobson, 1988). In Tahoe, the essence is to make the window size increase gradually until congestion is detected, and at congestion window size is reset to one and starts to increase it again. After that, plenty of studies were done to reach better performance based on this idea. Two famous ones are Reno (Jacobson, 1990) and Vegas (Brakmo and Peterson, 1995). Another approach by Chen *et al*. proposed an idea that uses the RTT to adjust the window size directly (Chen *et al*., 2000). Also in

(Srijith *et al*., 2005) and (Maor and Mansour, 2003), the predefined thresholds were dynamically adjusted to make TCP-Vegas more adaptive, and when congestion is detected the Window Size Incremented or Decremented in Steps. In Event of Link Failure the window establishment process should start from One again, called as Slow Start. In Event of Duplicate Acknowledgement the Window Size was dropped to Half of the Current Window and to continue the Window size recovery from there.

In 2010 Hou-Tsan Lee, Feng-Li Lian, and Ting-Chun Fong, proposed Bandwidth Based TCP, Ref. Journal of Chinese Institue of Engineers, Vol.33, No.3, pp. 489-494 (2010), in which the link Failure Recovery continued to be Modified Slow Start with Window Size reset to One as well as Duplicate Acknowledgement Recovery from Half of the Current Window size.

In Past few years the analysis has been continued to identify the better methods to enhance the performance of the TCP connectivity and the corresponding networks, this brings out the analysis done on recovery mechanism of various methods and the proposal shared in this paper work.

In this paper work the short fall of TCP Vegas and Bandwidth Based TCP has been discussed, and proposed, Memorized Window Consolidation (MWC) based TCP Recovery

II. TCP Vegas

The flow control in TCP is based on the Window Sized, i.e. destination responds with the acknowledgement for the packet's sent, and Source keeps increasing the Window size. Example of Windowing is shown in Fig.-1. In Event of Link disconnection the destination does not receive the packet and thus will not send any response to source, or the Source Does not receive the Acknowledgement from Destination. Detecting such link disconnect condition the source resets it Window Size to 1 and does a fresh Start over and slowly increase the Window Size step by step in a range of -1, 0 or 1, figure shows the behavior of TCP Vegas Window size in event of Link Disconnection.

If Duplicate acknowledgements are received to Source due to multiple path and delay in the network, TCP Vegas immediately decreased to the Window size to Half of the Current Window size, and enters the congestion avoidance, and does increased the window size step by Step, Figure shows the TCP-Vegas behavior in event of duplicate Acknowledgements

Figure 3 shows the Algorithm used by TCP -Vegas

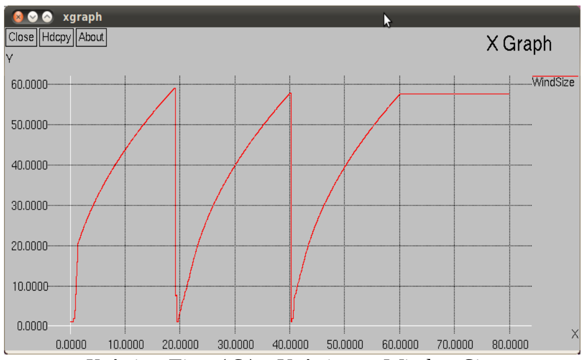
III. Bandwidth Based TCP

As an Enhancement to TCP Vegas, Bandwidth based TCP used the Epsilon and Current Window size as Window incremented decision maker, in Place of Alpha, Beta, Delta and Gama. And in Event of Link disconnection, it resets the Window size and Epsilon value to initial value of One. And does a fresh start over with these initial values continuing its Bandwidth estimation and increase the Window Size accordingly. Figure 4 shows Bandwidth Based TCP Behavior in event of link failure.

The Duplicate acknowledgement identification by Bandwidth based TCP make the window size Fifty percent of current window size, and then start over to increase the window based on new estimation and Epsilon value.

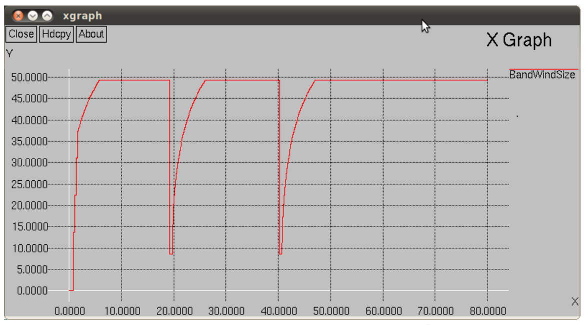
```
Slow-start
$W\_init \leftarrow 1$;
$delta \leftarrow (W/RTTmin - W/RTT) * RTTmin$;
for each ack
if (delta < gamma) then
$W \leftarrow W+1$;
else
enter Congestion Avoidance
end if

Fast Retransmit
if (dup_ACKs)
retransmit the lost packet
$W \leftarrow W/2$;
enter Congestion Avoidance
end if
```

Fig.2 Algorithm Used for TCP-Vegas



X-Axis Time (S) , Y-Axis Window Size

Fig.3 TCP Vegas Window at Link Recovery



X-Axis Time (S) , Y-Axis Window Size

Fig.4 B/w Based Window at Link Recovery

```
**Modified Slow-start for B/w based TCP**
$W\_init \leftarrow 1$;
$E\_init \leftarrow 2$;
for each ack
if ($RTT{=}{=}RTTmin$) then
$W \leftarrow W+1$;
$E \leftarrow E$;
else
enter Congestion Avoidance
end if

**Modified Fast Retransmit for B/w based TCP**

if (dup_ACKs) then
retransmit the lost packet
$W \leftarrow W/2$;
$E \leftarrow E/2$;
enter Congestion Avoidance
end if
```

Fig.5 Algorithm for Bandwidth Based TCP

IV. Memorized Window Consolidation (MWC) based TCP Recovery

As we have identified the problem in both Vegas and Bandwidth based, for the slow start, this sections discuss about the proposed solution. It has been identified that when the link is in a stable condition most of the time window size is approximately constant. For any recovered link the window will be increasing constantly from small value to a larger value. Also during initial stage when the link was established for the first time the window size will start from a lower value and will increase to a higher value. The window size will continue to grow till no congestion is observed and / or till the window size is stabilized by the algorithm or by a source sender. And when the link experiences a disconnection and reestablishes, we already had a good and stabilized window size in the past time period. Currently this value of past Stable history is not collected, in our approach we will be memorizing the past good values in to a memory, the memorization can be expressed as bellow

$\mathbf{F(Xn)_{n = 1 ----- 100}}$ **Eq. 1**

For Bandwidth Based we have additional Value of Epsilon so we will have additional Equation

$\mathbf{E(Yn)_{n = 1 ----- 100}}$ **Eq. 2**

Where the Xn is the current window size taken during each RTT, when the current window size is estimated and calculated. In this attempt we are considering last Hundred consecutive values of the window size, these values will be stored in a First in First out (FIFO) Fashion, which means as soon as we get next RTT and new window is decided we will be entering this value in memory and the oldest one will be discarded.
When the link is recovered from disconnection and then instead for Traditional Slow Start, we will look into the memory for last 100 window values and consider the 95th Percentile of these different values to be used, as a starting window size after recover in place of slow start. When implemented this algorithm, the very first slow start will not be impacted as all the memory elements will be initially set to One so the very first slow start will start from a window size of One as usual. While using the above algorithm alone we identified that, it may happen to have a good Window size at initial stages and due to increase of traffic between different source and destination, the current or the latest window size might have been decreased, so considering the Hundred iterations only and its 95th Percentile may not give a very accurate calculation. So we have also considered the latest last 5 values of the Window size prior the link was disconnected and can be given as Bellow

$\mathbf{F(XT\text{-}t)_{t=1-5}}$ **Eq. -3**

$\mathbf{E(YT\text{-}t)_{t=1-5}}$ **Eq. -4**

Where T is last iteration F(Xn)

Now as we have the 100 iterative values and the latest last 5 best values, we will check if the 95th percentile of 100 values or the least value of the last latest 5 values, have the lower window size and correspondingly we will consider the least of both values as the starting window size after recover of the link and can be given as bellow

MWC Based Slow-start for TCP Vegas

```
if  least of $F(XT\text{-}t)_{t=1-5}$
            < $95^{th}$ % $F(Xn)_{n = 1 ----- 100}$

    then   W_init = $F(XT\text{-}t)_{t=1-5}$
else
    W_init = $95^{th}$ % $F(Xn)_{n = 1 ----- 100}$

                    ------------ Eq. -5
```

MWC Based Fast Retransmit-TCP Vegas

```
if (dup_ACKs) then
retransmit the lost packet

If  least of $F(XT\text{-}t)_{t=1-5}$
          < $95^{th}$ % $F(Xn)_{n = 1 ----- 100}$

    then   W = $F(XT\text{-}t)_{t=1-5}$
else
        W = $95^{th}$ % $F(Xn)_{n = 1 ----- 100}$

enter Congestion Avoidance
end if                    ------------ Eq. -6
```

Fig.6 MWC Algorithm for TCP Vegas

MWC Based Slow-start-B/W Based TCP

```
If  least  of  $F(XT\text{-}t)_{t=1-5}$
                 $< 95^{th}\ \%\ \ F(Xn)_{n=1 ----- 100}$
      then  {   $W\_init = F(XT\text{-}t)_{t=1-5}$
            $E = E(YT\text{-}t)_{t=1-5}$ }
   else   {  $W\_init = 95^{th}\ \%\ \ F(Xn)_{n=1 ----- 100}$
            $E = 95^{th}\ \%\ \ E(Yn)_{n=1 ----- 100}$ }
                                 ------------- Eq. -7
```

MWC Fast Retransmit-B/W Based TCP

```
if (dup_ACKs) then
retransmit the lost packet

If  least  of  $F(XT\text{-}t)_{t=1-5}$
                 $< 95^{th}\ \%\ \ F(Xn)_{n=1 ----- 100}$
      then   { $W = F(XT\text{-}t)_{t=1-5}$
            $E = E(YT\text{-}t)_{t=1-5}$ }
         else   { $W = 95^{th}\ \%\ \ F(Xn)_{n=1 ----- 100}$
            $E = 95^{th}\ \%\ \ E(Yn)_{n=1 ----- 100}$ }

enter Congestion Avoidance
end if
                                 ------------- Eq. -9
```

Fig.7 MWC Algorithm B/w Based TCP

V. Simulation Results

Post presenting the design in Section III, Simulations via NS-2 (The Network Simulation – Version 2) are demonstrated in this section under environment as bellow, which will help to judge the performance of the TCP during recovery. Here we have taken two different scenarios, as shown bellow, the location shows the link disconnection used in simulation to identify the test results, at the time shown in timing diagram. These simulations were tested both for TCP Vegas and Bandwidth Based TCP. Scenario 1consist of a Source and Destination connecting via 3 Routers in between, links connected using 1 Mbps bandwidth with a delay of 10 ms. Scenario 2 consist of a higher bandwidth between Source and Destination to router and , but the link between routers are equipped with lesser bandwidth of 1Mb and a higher latency of 30 ms. It has been identified that the Window recovery Mechanism has given the similar results for increment of the Window size, even though the Queue size varies depending upon the scenario, which is not displayed here due focus on the recovery of window size. Each Wave form output shows the results with X-Axis of Time and Y-Axis of Window size in Packets/ Sec, for the time intervals when the link was up, Down, and during the Window Recovery

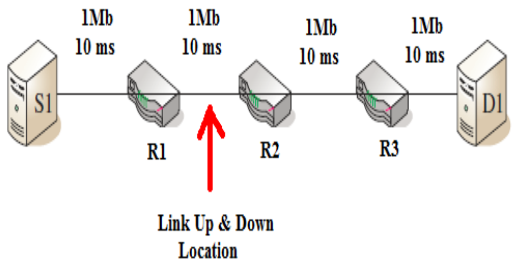


Fig.8 One Sender Configuration Scenario 1

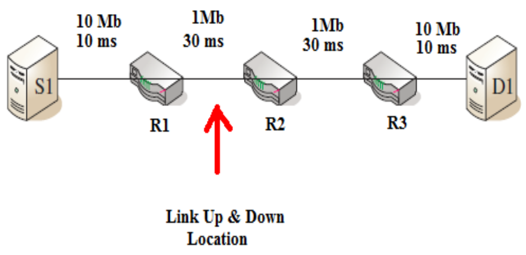
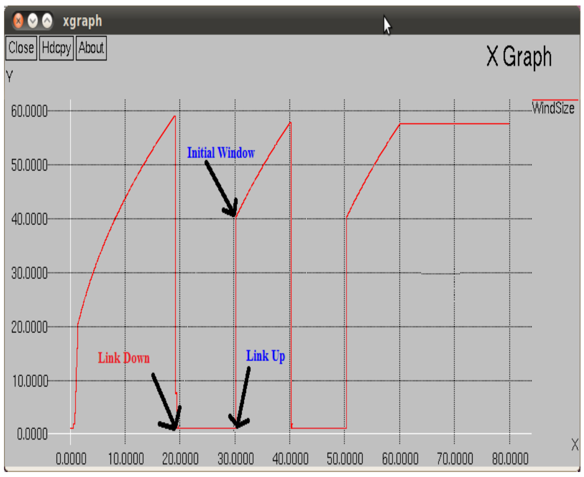
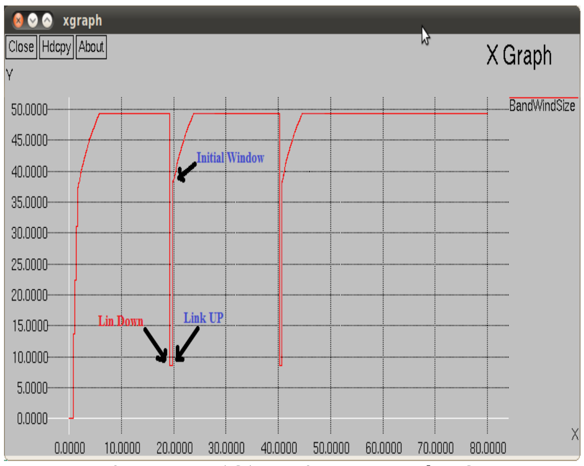


Fig.9 One Sender Configuration Scenario 2



X-Axis Time (S) , Y-Axis Window Size

Fig. 10 MWC based Window for TCP-Vegas



X-Axis Time (S) , Y-Axis Window Size

Fig. 11 MWC based Window for B/w based TCP

VI. Conclusion

In this paper, the Memorized Window based TCP window recovery is proposed. It has Three Aspects : Memorize Window Size, Check Latest Five Windows, Start from Best Window Size prior Link Down. In brief it efforts to the quick window recovery in event of link recover, thus enhancing the overall through put of connection, making better connectivity. Through the simulations, the proposed scheme is shown to have better performance than Vegas and B/w Based TCP under homogeneous and a heterogeneous environment.

Though the MWC is superior to TCP Vegas and B/w based TCP, some aspects can be considered for future analysis, such as if Receiver is disconnected the initial window judgment should be seen considering rest of active connection to other destination from same source, which will be tackled in future.

VII. Nomenclature

W	Window size of the Packet / Sec
E	Epsilon Window Incr. value in B./w based TCP
delta	TCP-Vegas differential identifier
RTTmin	TCP Packet min Round Trip Time
RTT	TCP Packet Current RTT
T	Last Window Iteration Number

References

[1] Hou-Tsan-Lee, Feng-Li Lian, and Ting-Chun Fong, “Congestion Control of TCP based on Bandwidth Estimation”, Journal of the Chinese Institute of Engineers Vol. 33, No. 3. Pp, 489-494 (2010)

[2] RFC-2581 M.Allman, NASA Glenn/Sterling Software, V.Pasxon, ACIRI / ICSI, W.Stevens Consultant “ https://tools.ietf.org/rfc/rfc2581.txt”

[3] Brakmo, L. S., and Peterson, L. L., 1995, “TCP Vegas: End to End Congestion Avoidance on a GlobalInternet,” *IEEE Journal on Selected Areas in Communications*,Vol. 13, No. 8, pp. 1465-1480.

[4] Chen, J. R., Chen, Y. C., and Lee, C. L., 2000, “TCP Vegas-A: Improving the Performance of TCP Vegas,” *Computer Communications*, Vol. 23, No. 16, pp. 1537-1547.

[5] Floyd, S., and Jacobson, V., 1993, “Random Early Detection Gateways for Congestion avoidance,” *IEEE /ACM Transactions on Networking*, Vol. 1, No. 4, pp. 397-413.

[6] Gerla, M., Sanadidi, M. Y., Wang, Ren, Zanella, A., Casetti, C., and Mascolo, S., 2001, “TCP Westwood: ongestion window control using bandwidth estimation,” *IEEE GLOBECOM ’01*, Vol. 3, pp. 1698-1702.

[7] Hollot, C. V., Misra, V., Towsley, D., and Gong, W. B., 2001, “On Designing Improved Controllers for AQM Routers Supporting TCP Flows,” *in Proceedings of IEEE INFOCOM*, Vol. 3, pp. 1726-1734, Alaska, USA.

[8] Hollot, C. V., and Chait, Y., 2001, “Nonlinear Stability Analysis for a Class of TCP/AQM Networks,” *in Proceedings of IEEE CDC*, Vol. 3, pp. 2309- 2314, Orlando, FL, USA.

[9] Jacobson, V., 1988, “Congestion Avoidance and Control,” *in Proceedings of ACM SIGCOMM*, pp. 314-329, Stanford, CA, USA.

[10] Jacobson, V., 1990, “Berkeley TCP Evolution from 4. 3-Tahoe to 4.3-TCP-Reno,” *Proceedings of the 18th Internet Engineering Task Force*, Vancouver, BC, Canada.

[11] Low, S. H., Paganini, F., and Doyle, J. C., 2002, “Internet Congestion Control,” *IEEE Control Systems Magazine*, Vol. 22, No. 1, pp. 28-43.

[12] Moar, A., and Mansour, Y., 2003, “AdaVegas: Adaptive Control for TCP Vegas,” *Proceedings of the IEEE GLOBECOM’03*, Vol. 7, pp. 3647-3651.

[13] Srijith, K. N., Jacob, L., and Ananda, A. L., 2005, “An End-to-End Flow Control Approach Based on Round Trip Time,” *Computer Communication*s, Vol. 28, No. 4, pp. 429-440.

Mr. Shree Ammu Graduate in Electronics and Communications from Priydarshani College of Engineer and Architecture, Nagpur University
Diploma in Electronics and Telecommunications from Anjuman Polytechnic, Bombay Technical BoardCCIEQ-Voice, CCNP-Route,Switch, Checkpoint--CCSA ISMS-LA-27001, Dip. In Cyber Law, Now pursuing Masters in **Digital Electronics and Communication Systems** (DECS) from Sri Indu College of Engineering & Technology.I express my gratitude to

Prof. Suresh Ballala Department (ECE) for his constant co-operation, support and for providing necessary facilities throughout the M.tech program. He have more than 15 Years of Teaching Experience, at B.Tech and M.tech, Level which includes Microprocessors, Microcontrollers, Core-Jave etc. and working as a Professor in Sri Indu College of Engg.& Technology Affiliated to JNTU Hyderabad.