Advanced Science and Technology Letters
Vol.31 (MulGraB 2013), pp.126-129
http://dx.doi.org/10.14257/astl.2013.31.28

Delay-based Congestion Control for Multipath TCP

Wenlan Guo[1], Zhijia Wang[1], Yun Zhang[2]

[1] School of Computer Science and Technology,Harbin Univ.Sci.Tech.,Harbin,China
guowenlan@sohu.com
[2] Department of Electrical and Computer Engineering,University of Texas at Austin
zhangyun@utexas.edu

Abstract. A single flow stream could be divided into several sub-flows to be sent across multiple paths. This method obtains obvious advantages against traditional TCP as it maintains higher reliability and makes better use of the network resources. Congestion control algorithm is an essential part to be considered. This paper first reviews the existing multipath TCP congestion control algorithms and then analyzes and formulates the goals and problems need to be achieved and solved. A delay-based congestion control algorithm named Weighted Vegas (wVegas) is provided. Finally, two possible modifications are demonstrated including adjusting the congestion control window according to how far the path is from congestion and redefining the behavior when loss occurs.

Keywords: Congestion control; Multipath TCP; Weighted Vegas

1 Introduction

There are three goals during the design of multipath TCP congestion control algorithm including efficiency, fairness and the congestion depended utilization of each available path. A number of multipath TCP algorithms have already been proposed. The uncoupled algorithms apply congestion control on each path independently. This kind of algorithms is difficult to ensure the fairness, like Ptcp[1], , CMT-SCTP [2] and M/TCP [3]. Some uncoupled methods like EWTCP [4] applies weight on each sub-flow could ensure fairness, but it could hardly make full use of network resource. The fully coupled algorithms will couple the congestion control on each involved path. However it leads to the result that the traffic would totally shift to the less congested flow which would sometimes causes link to shut down. The robust algorithm always applies semi-coupled method in which the traffic will prefer the less congested paths while keep a reasonable traffic on the congested ones, such as MPTCP [4].

Most of current multipath TCP congestion control algorithms use the loss of packets as the indication of the congestion. It provides congestion detection, but not the avoidance. Network resources have been wasted if the congestion control starts to shift the traffic when the loss occurs. One more problem is the buffer overflow loss in the loss-based congestion control as the sending window would increase infinitely if

ISSN: 2287-1233 ASTL

no loss happens. Therefore another approach is proposed in the following sections which use delay to indicate extension of the congestion.

2 Weighted Vegas Congestion Control Algorithm

According to the formulated optimization goal of congestion control [5], our Weighted Vegas Congestion Control Algorithm could be derived. Given that the BASE_RTT presents the minimum RTT has been measured so far. RTT presents the average RTT during the previous round. The CWND presents the size of the congestion control window. The DIFF presents for how many packages are backlogged in the queue.

$$DIFF = \frac{CWND \cdot \frac{CWND}{BASE_{RTT}}}{BASE_{RTT}} = \frac{CWND(RTT - BASE_{RTT})}{RTT} \quad (1)$$

The DIFF could be defined the backlogged package, the price of the path expresses as $q_r = RTT - BASE\_RTT$ and the transmission rate is expressed as $\frac{CWND}{RTT}$.

Assuming the network holds a set of L of physical links with the finite capacity c_l (l belongs L). A path r (belongs to R) holds a set of link L_r. $a_{l,r}$ could be used to presents the relationship between the paths and the links. If l belongs to the set of links belongs to path L_r, then the $a_{l,r} = 1$, otherwise it stays zero. Each flow s belongs to a set of flow S would hold a set of path R_s. $b_{s,r}$ could be used to presents the relationship between the paths and the flows. If path r is one of the transmission tube of flow s (r belongs to R_s), then the $b_{s,r} = 1$, otherwise it would stay zero. $x_{s,r}$ presents the transmission rate of flow s on the path r.

$$x_{s,r} = \frac{a_{s,r}}{q_r} \quad r \in R_s \quad ()$$

When the network reaches the optimal situation, the price of each path approaches the equal value, say q_s. And the total rate of a flow could be expressed as $y_s = \frac{a_s}{q_s}$ where the a_s represents the total backlogged packages over all path for a flow. And in the optimal situation, the expected price $U_s' = \sum_{r \in R_s} x_{s,r}' = q_s = \frac{a_s}{y_s}$ (proved in [1]) and the utility function could be obtained as $U(y_s) = a_s \log y_s$. Then assign the

increasing step factor $\theta = x_{s,r(n)}$ q_r
could be obtained as below.

$$x_{s,r}(n+1) = \frac{x_{s,r}(n)}{y_s} \quad \frac{a_s}{q_r} \quad k_{s,r}(n)a_s$$

$k_{s,r}(n) = \frac{x_{s,r}(n)}{y_s}$ is expressed as the weight of flow s on the path r. From the above

expected backlogged packages are treated as the weighted V(q) as in this algorithm.

. The optimal transmission rate for one itineration

3 Implementation of Weighted Vegas

Then it comes a special implementation of the algorithm. it gives a variable array **equilibrium_rates[r]** to store the transmission rate of path r, a **queue_delay[r]** to store the minimum queue delay of path r and **alpha[r]** to store the expected backlogged packets. The pseudo code and the analysis of the specific algorithm are shown as below.

In the Initialization phase:

- **total_alpha** <- **fixed number----** The network would assign a fixed number of backlogged packages to the coming flow and save it in the total_alpha.
- **Equilibrium_rates[r]** <- **0, queue_delay[r]** <- **0, alpha[r] <-random number**
- **Base_RTT[r]----Measure** the minimum RTT of path with no queuing delay

During the congestion avoidance phase after each round of path r:

- RTT <- **Sampled_rtts[r]/Sampled_num[r]----** Calculate the average RTT of the previous round
- **Diff** <- **cwnd[r] (RTT-Base_RTT[r])** / RTT----Calculate the actual number of backlogged packages of path r. It is the queuing delay indicates extension of congestion
- **If Diff>=alpha[r] then**
- **Equilibrium_rates[r]<-cwnd[r]/RTT**
- **Adjust weight[r]<-Equilibrium_rates[r]/total_rate**
- **alpha[r]<-weight[r] multiply total_alpha**
- **alpha[r]<-max (alpha[r], 2) end if**

----This if block is used for updating the transmission rate, weight and expected backlogged packages of path r. The transmission rate and the weight is only updated when the number of actual backlogged packages is larger than the number of expected backlogged packages. The last instruction ensures that even if the path occupies no weight, it is still not abandoned and assigned a reasonable load.

- **If Diff<alpha[r] then cwnd[r] <- cwnd[r]+1**
- If Diff>alpha[r] then cwnd[r] <- cwnd[r]-1

----Update the sending window size according to the relationship between expected backlogged packages and actual backlogged packages.

- **q<-RTT** - Base_RTT
- if queue_delay[r]=0 or queue_delay[r]>q then **queue_delay[r]** <- **q**
- if q>2 multiply **queue_delay[r]**
- backoff_factor <- **0.5 multiply Base_RTT[r]/RTT**
- **queue_delay[r]<-0**
- **cwnd[r]** <- max (cwnd[r], 2)---Ensure that no path is abandoned totally

If package loss occurs on path r:

equilibrium_rates[r] <- **0, queue_delay[r]** <- **0----**When loss occurs, the path is abandoned. All the traffic is shifted to other paths.

4 Modification and Problem

One problem lies on the accuracy of the RTT measurement. When the arriving rate is much higher than the transmission rate, it could hardly be able to obtain the accurate RTT. The arriving rate is a limitation to the Weighted Vegas algorithm. A possible modification could be made on how to change the congestion control window according to the queuing delay. Actually the queuing delay could do more than just judging whether the path is congested or not, it could also tell how far it is from the situation of congestion. When it is still far from the congestion, the congestion window could be expanded greatly during each round. As it approaches the congestion situation, the increase of the sending window should slow down. This idea could obvious accelerate the process of converging. Another modification could be made on the behavior when loss occurs. The error in the package will also lead to a loss causing avoidance of path. This will waste the bandwidth if the loss is not due to congestion. To obtain a better performance, it should determine the action according to the situation of congestion rather than just abandoning the path when loss package occurs.

5 Conclusion

Based on the Lagrangian optimization of the congestion control goal., we proposed a delay based iterative congestion control algorithm. A specific implementation of this algorithm is also discussed. After implementing proposed algorithm in MATLAB and comparing with other methods like MPTCP, this Weighted Vegas algorithm obviously ensures the fairness and efficiency requirement and could also better balance a dynamic traffic. The limitation on the input packets rate and two possible modifications to improve the performance are also discussed.

References

1. H.-Y. Hsieh and R. Sivakumar. A transport layer approach for achieving aggregate bandwidths on multi-homed mobile hosts. In Proc. MobiCom ’02, pp. 83--94, New York, NY, USA, ACM(2002)
2. J. R. Iyengar, P. D. Amer, and R. Stewart. Concurrent multipath transfer using SCTP multihoming over independent end-to-end paths. IEEE/ACM Trans. Netw., 14(5):951--964, (2006)
3. K. Rojviboonchai and H. Aida. An evaluation of multi-path transmission control protocol (M/TCP) with robust acknowledgement schemes. IEICE Trans. Communications(2004)
4. Damon Wischik, Costin Raiciu, Adam Greenhalgh and Mark Handley. Implementation and Evaluation of Congestion Control for Multipath TCP. NSDI(2011)
5. Yu Cao, Mingwei Xu, Xiaoming Fu. Delay-based Congestion Control for Multipath TCP. ICNP. 1--10(2012)

