Advances in Congestion Control Algorithms in Internet

Jubilant J Kizhakkethottam[1], Vinod Chandra S S[2], Jayasudha J S[3]

[1]Department of Computer Science, St. Joseph College of Engineering & Technology, Pala
[2]Department of Computer Applications, College of Engineering Trivandrum
[3]Department of Computer Science, SCT College of Engineering, Thiruvananthapuram

Abstract

The review is done focused on Transmission Control Protocol/Internet Protocol (TCP/IP) by studying almost recently proposed algorithms to avoid congestion and improve performance. Our study in this work list many open issues that persevere in the designing, functioning and managing of the Internet. It aims at preventing incoming packet drops at gateways because of local buffer overflows. We identified that the enormous part of problems lies in Transport Protocol Implementation which will never mean it happens in Protocols themselves which may sometimes happen.

INTRODUCTION

End-to-end congestion control mechanisms of TCP was been a significant feature in the vigor of the internet. Still, the Internet is no longer practical to rely on all end-nodes to use end-to-end congestion control for best- effort traffic. It is no longer possible to rely on all developers to incorporate end-to-end congestion control in their Internet applications. The network itself must now participate in controlling its own resource utilization. Assuming the Internet will continue to become congested due to a scarcity of bandwidth, this proposition leads to several possible approaches for controlling best-effort traffic.[45]One approach involves the deployment of packet scheduling disciplines in routers that isolate each flow, as much as possible, from the effects of other flows. This approach suggests the deployment of per-flow scheduling mechanisms that separately regulate the bandwidth used by each best-effort flow, usually in an effort to approximate max-min fairness.

A second approach, outlined in this paper, is for routers to support the continued use of end-to-end congestion control as the primary mechanism for best-effort traffic to share scarce bandwidth, and to deploy incentives for its continued use. These incentives would be in the form of router mechanisms to restrict the bandwidth of best-effort flows using a disproportionate share of the bandwidth in times of congestion. These mechanisms would give a concrete incentive to end users, application developers, and protocol designers to use end-to-end congestion control for best-effort traffic.

Another approach in this paper would be to rely on financial pricing mechanisms to control sharing. Relying exclusively financial incentives would result in a risky gamble that network providers will be able to provision additional bandwidth and deploy effective pricing structures fast enough to keep up with the growth in unresponsive best-effort traffic in the Internet. These three approaches to sharing, of per-flow scheduling, incentives for end-to-end congestion control, and pricing mechanisms, are not necessarily mutually exclusive. Given the fundamental heterogeneity of the Internet, there is no requirement that all routers or all service providers follow precisely the same approach.

Present TCP Protocols

TCP uses “window” flow control,[11] where a destination sends acknowledgments for packets that are correctly received. A source keeps a variable called window size that determines the maximum number of outstanding packets that have been transmitted but not yet acknowledged. When the window size is exhausted, the source must wait for an acknowledgment before sending a new packet. Two features are important. The first is the “self-clocking” feature that automatically slows down the source when a network becomes congested and acknowledgments are delayed. The second is that the window size controls the source rate: roughly one window of packets is sent every round-trip time. The first feature was the only congestion control mechanism in the Internet before Van Jacobson’s proposal in 1988 [24]. Jacobson’s idea is to dynamically adapt window size to network congestion. In this section, we will review how TCP infers congestion and adjusts window size. TCP also provides other end-to-end services such as error recovery and round-trip time estimation, but we will limit our attention to the congestion control aspect.

TCP Vegas

TCP Vegas improves [12] upon TCP Reno through three main techniques. The first is a new retransmission mechanism where timeout is checked on receiving the first duplicate acknowledgment, rather than waiting for the third duplicate acknowledgment (as Reno would), and results in a more timely detection of loss. The second technique is a more prudent way to grow the window size during the initial use of slow-start when a connection starts up, and it results in fewer losses. The third technique is a new congestion avoidance mechanism that corrects the oscillatory behavior of Reno. The idea is to have a source estimate the number of its own packets buffered in the path and try to keep this number between α (typically 1) and β (typically 3) by adjusting its window size. The window size is increased or decreased linearly in the next round-trip time according to whether the current estimate is less than α or greater than β. Otherwise the window size is unchanged. The rationale behind this is to maintain a small number of packets in the pipe to take advantage of extra capacity when it becomes available. Another interpretation of the congestion avoidance algorithm of Vegas is given in [12], in which a Vegas source periodically measures the round-trip queuing delay and sets its rate to be proportional to the ratio of its round-trip propagation delay to queuing delay, the proportionality constant being between α and β. Hence, the more congested its path, the higher the queuing delay and the lower the rate. The Vegas source obtains queuing delay by monitoring its round-trip time (the time between sending a packet and receiving its acknowledgment) and subtracting from it the round-trip propagation delay.

New Westwood TCP

Overview of the algorithm New Westwood TCP is a sender side only implementation of the MEAD mechanism that follows the fundamental end-to-end Internet design principle [4,5]. The key idea is to exploit the flow of returning ACKs to estimate both available bandwidth and queue backlog. NW TCP preserves the standard TCP slow start and congestion avoidance phases in order to probe the network capacity until congestion is experienced. While in standard TCP (i.e. Tahoe Reno) congestion is signaled by timeouts or duplicate acknowledgments, in NW TCP the source becomes aware of congestion by estimating the queue backlog. This allows the sender to (1) promptly detect congestion before queue overflow, (2) increase the fairness since each sender sets the same backlog threshold for each flow, (3) discriminate congestion from losses due to unreliable links. In order to estimate the queue backlog, NW TCP needs to estimate the available bandwidth and the queuing time. [21]The queuing time is measured by time-stamping packets and by subtracting the minimum measured round trip time.

The end-toad estimate of the available bandwidth is obtained by low-pass filtering the rate of returning ACKs. The estimate is then multiplied by the queuing time to obtain an estimate of the queue backlog. When the queue backlog is greater than a threshold then the TCP sender sets the congestion window and the slow start threshold equal to the available bandwidth times the minimum round trip time. [41]The rationale of this strategy is simple: in contrast with TCP Reno, which implements a "blind" multiplicative decrease algorithm after congestion, NW TCP adaptively sets a slow start threshold and congestion windows, which are consistent with the bandwidth used at the time congestion is detected.

RED

RED (random early detection) [35,36] is an alternative way to generate the congestion measure (loss) to Reno sources. Instead of dropping only at a full buffer, RED maintains an exponentially weighted queue length and drops packets with a probability that increases with the average queue length. When the average queue length is less than a minimum threshold, no packets are dropped. When it exceeds a maximum threshold, all packets are dropped. When it is in between, a packet is dropped with a probability that is a piecewise linear and increasing function of the average queue length. This type of strategy is called active queue management (AQM).

FIFO

A Vegas source adjusts its rate based on observed queuing delay; in other words, it uses queuing delay as a measure of congestion. [40]This information is updated by the FIFO (first-in-first-out) buffer process and fed back implicitly to sources through round-trip time measurement.

Droptail

A [37] Reno source uses loss as a measure of congestion. This information is typically generated and fed back to sources through Droptail, a queuing discipline that drops an arrival to a full buffer.

TCP Reno/RED

We focus only on the congestion avoidance phase of TCP Reno, in which an elephant typically spends most of its time.[38] We take source rates as the primal variable x and link loss probabilities as prices p. In this section, we assume the round-trip time ti of source i is constant and that rate xi, is related to window wi.

ACC for TCP

As a test of the ACC principles outlined above, we have defined an active congestion control based on TCP. TCP contains a classic, well understood feedback control system: the congestion avoidance mechanisms defined by Jacobson [5]. Endpoint sending rate is controlled by a sliding window which is advanced by packet acknowledgments. The size of the window is modulated in response to congestion along the connection’s path. The window modulation algorithm in TCP is a classic linear increase/multiplicative decrease algorithm. When congestion is detected, the window is reduced to half its current size. When a full window of consecutive packets has been acknowledged without congestion being detected, the window is increased by one maximum-sized packet. We omit the discussion of the Slow-Start algorithm because the current work considers primarily steady-state effects.

Fairness

It is well known that TCP Reno discriminates against connections with large propagation delays. This is clear from [17], which implies that Reno equalizes windows for sources that experience the same loss probability, and hence their rates are inversely proportional to their round-trip times. The equilibrium characterization [17] also exposes the“beat down” effect, where sources that go through more congested links, seeing larger qi, receive less bandwidth.

This effect is hidden in single-link models and, in multilink models, is often confused with delay-induced discrimination of TCP, as expressed in (17). It has been observed in simulations [44] and has long been deemed unfair, but the duality model shows that it is an unavoidable and even desirable, feature of end-to-end congestion control. For each unit of increment in aggregate utility, a source with a longer path consumes more resources and hence should be beaten down. If this is undesirable, it can be remedied by weighting the utility function with delay.

Delay and Loss

The [28,29]current protocol (Reno with DropTail) fills, rather than empties, bottleneck queues when the number of elephants becomes large, leading to a high loss rate and queuing de- lay. What is more intriguing is that increasing the buffer size does not reduce loss rate significantly, but only increases queuing delay. This delay and loss behavior is exactly opposite the mice-elephant control strategy we aim for: to maximally utilize the network in a way that leaves network queues small so that delay-sensitive mice can fly through the network with little queuing delay.

According to the duality model, loss probability under Reno is the Lagrange multiplier, and hence its equilibrium value is determined solely by the network topology and the number of sources, independent of link algorithms and buffer size. Increasing the buffer size but leaving everything else unchanged does not change the equilibrium loss probability, and hence a larger backlog must be maintained to generate the same loss probability. This means that with DropTail, the buffer at a bottleneck link is always close to full, regardless of buffer size. With RED, since loss probability is increasing in average queue length, the queue length must increase steadily as the number of sources grows.

Conclusions

This work has focused on describing various congestion control mechanisms used in Internet using TCP/IP and a study of fairness, delay and loss which affects in transmission. All the variants used in the paper are good in some aspects and avoids some problems. As the Internet become more popular it has to improve its performance to meet the requirements. Algorithms and techniques has to be more efficient and fair to meet the needs. Delay and loss rate has to be minimum and fairness and quality has to be at its highest for meeting the needs.

References

[1] L. Benmohamed and S.M. Meerkov, “Feedback control of congestion in store-and-forward networks: The case of a single congested node,” IEEE/ACM Trans. Networking, vol. 1, pp. 693-707, Dec. 1993.

[2] S. Chong, R. Nagarajan, and Y.-T. Wang, “Designing stable ABR flow control with rate feedback and open loop control: First order control case,” Perform. Eval., vol. 34, no. 4, pp. 189-206, Dec. 1998.

[3] E. Altman, T. Basar, and R. Srikant, “Congestion control as a stochas- tic control problem with action delays,” Automatica, Dec. 1999.

[4] H. Ozbay, S. Kalyanaraman, and A. Iftar, “On rate-based congestion control in high-speed networks: Design of single bottleneck,” in Proc. Amer. Control Conf., 1998.

[5] S. Mascolo, “Congestion control in high-speed communication net- works using the Smith principle,” Automatica, vol. 35, no. 12, pp. 1921-1935, Dec. 1999.

[6] E.J. Hernandez-Valencia, L. Benmohamed, R. Nagarajan, and S. Chong, “Rate control algorithms for the ATM ABR service,” Eur. Trans. Telecommun., vol. 8, pp. 7-20, 1997.

42 based flow controller for

[7] R. Srikant, “Control of communication networks,” in Perspectives in Control Engineering: Technologies, Applications, New Directions,T. Samad, Ed. Piscataway, NJ: IEEE Press, 2000, pp. 462-488.

[8] F.P. Kelly, A. Maulloo, and D. Tan, “Rate control for communication networks: Shadow prices, proportional fairness and stability,” J. Oper. Res. Soc., vol. 49, no. 3, pp. 237-252, Mar. 1998.

[9] F.P. Kelly. (July 1999). Mathematical modelling of the Internet, in Proc. 4th Int. Congr. Industrial Applied Mathematics. Available: http:// www.statslab.cam.ac.uk/~frank/mmi.html

[10] S.H. Low and D.E. Lapsley, “Optimization flow control, I: Basic algo- rithm and convergence,” IEEE/ACM Trans. Networking,vol.7,pp. 861-874, Dec. 1999. Available: http://netlab.caltech.edu

[11] S.H.Low. (Sept. 18-20, 2000).AdualitymodelofTCPflow controls, in Proc. ITC Specialist Seminar on IP Traffic Measurement, Modeling and Management. Available: http://netlab.caltech.edu

[12] S.H. Low, L. Peterson, and L.Wang. (June 2001). Understanding Ve- gas: A duality model, in Proc. ACM Sigmetrics. Available: http://netlab.caltech.edu/ pub.html

[13] S. Athuraliya and S.H. Low, “Optimization flow control with New- ton-like algorithm,” J. Telecommun. Syst., vol. 15, no. 3/4, pp. 345-358, 2000.

[14] F. Paganini. (2001).Onthe stability of optimization-based flow control, in Proc. Amer. Control Conf. Available: http://www.ee.ucla.edu/-paganini/ PS/remproof.ps

[15] S. Kunniyur and R. Srikant. (Mar. 2000). End-to-end congestion con- trol schemes: Utility functions, random losses and ECN marks, in Proc. IEEE Infocom. Available: http://www.ieee-infocom.org/2000/pa- pers/401.ps

[16] S. Kunniyur and R. Srikant. (Apr. 2001). A time-scale decomposition approach to adaptive ECN marking, in Proc. IEEE Infocom. Available: http:// comm.csl.uiuc.edu:80/~srikant/pub.html

[17] J. Mo and J. Walrand, “Fair end-to-end window-based congestion control,” IEEE/ACM Trans. Networking, vol. 8, no. 5, pp. 556-567, Oct. 2000.

[18] J. Mo, R. La, V. Anantharam, and J.Walrand, “Analysis and compari- son of TCP Reno and Vegas,” in Proc. IEEE Infocom. Mar. 1999.

[19] R. La andV. Anantharam. (Mar. 2000). Charge-sensitiveTCPand rate control in the Internet, in Proc. IEEE Infocom. Available: http://www.ieee-infocom.org/2000/papers/401.ps

[20] K. Kar, S. Sarkar, and L. Tassiulas, “Optimization based rate control for multirate multicast sessions,” in Proc. IEEE Infocom., Apr. 2001.

[21] V. Misra,W.-B Gong, and D. Towsley, “Fluid-based analysis of a net- work of AQM routers supporting tcp flows with an application to RED,” in Proc. ACM SIGCOMM, 2000.

[22] C. Hollot,V. Misra,D.Towsley, andW.-B. Gong. (Apr. 2001).Acontrol theoretic analysis of RED, in Proc. IEEE Infocom. Available: http://www-net.cs. umass.edu/papers/papers.html

[23] F. Paganini, “Flow control via pricing: a feedback perspective,” in Proc. 2000 Allerton Conf., Oct. 2000.

[24] V. Jacobson. (Aug. 1988). Congestion avoidance and control, in Proc. SIGCOMM’88, ACM. Available: ftp://ftp.ee.lbl.gov/papers/congavoid.ps.Z

[25] F.P. Kelly. (Dec. 1999). Models for a self-managed Internet, in Proc. R. Soc. Meeting. Available: http://www.statslab.cam.ac.uk/~frank/smi.html

[26] R. Johari and D. Tan, “End-to-end congestion control for the Internet: Delays and stability,” Cambridge Univ., Cambridge, U.K., Cam- bridge Univ. Statistical Laboratory Research Report, Tech. Rep. 2000-2, 2000.

[27] L. Massoulie, “Stability of distributed congestion control with heter- ogeneous feedback delays,” Microsoft Research, Cambridge, U.K., Tech. Rep. TR 2000-111, 2000.

[28] G.Vinnicombe, “On the stability of end-to-end congestion control for the Internet,” Cambridge Univ., Cambridge, U.K., Tech. Rep. CUED/ F-INFENG/TR.398, Dec. 2000.

[29]W.E. Leland, M.S.Taqqu,W.Willinger,andD.V.Wilson,“Onthe self-similar natureof Ethernet traffic,”IEEE/ACMTrans. Networking, vol. 2, pp. 1-15, 1994.

[30] V. Paxson and S. Floyd, “Wide-area traffic: The failure of Poisson model- ing,” IEEE/ACM Trans. Networking, vol. 3, pp. 226-244, 1995.

[31] W. Willinger, M.S. Taqqu, R. Sherman, and D.V. Wilson, “Self-similarity through high variability: Statistical analysis of Ethernet LAN traffic at the source level,” IEEE/ACM Trans. Networking, vol. 5, pp. 71-86, 1997.

[32] M.E. Crovella and A. Bestavros, “Self-similarity inWorldWideWebtraffic: Evidence and possible causes,” IEEE/ACM Trans. Networking, vol. 5, pp. 835-846, 1997.

[33] X. Zhu, J.Yu,andJ.C. Doyle,“Heavytails, generalized coding,andoptimal web layout,” in Proc. IEEE Infocom., Apr. 2001.

[34] L.S. Brakmo and L.L. Peterson, “TCPVegas: End to end congestion avoid- ance on a global Internet,” IEEE J. Select. Areas Commun., vol. 13, pp. 1465-1480, Oct. 1995. Available: http://cs.princeton.edu/nsg/papers/ jsac-vegas.ps

[35] S. Floyd and V. Jacobson, “Random early detection gateways for conges- tion avoidance,” IEEE/ACM Trans. Networking, vol. 1, pp. 397-413, Aug. 1993. Available: ftp://ftp.ee.lbl.gov/papers/early.ps.gz

[36] G. de Veciana, T.J. Lee, and T. Konstantopoulos, “Stability and perfor- mance analysis of networks supporting elastic services,” IEEE/ACM Trans. Networking, vol. 9 , Feb. 2001.

[37] F. BaccelliandD. Hong, “AIMD, fairnessandfractal scaling ofTCPtraffic,” INRIA, Paris, France, Tech. Rep. RR 4155, 2001.

[39] H. Yaiche, R.R. Mazumdar, and C. Rosenberg, “A game theoretic frame- work for bandwidth allocation and pricing in broadband networks,” IEEE/ACM Trans. Networking, vol. 8, pp. 2-14, Oct. 2000.

[40] T.V. Lakshman and U. Madhow, “The performance of TCP/IP for networks with high bandwidth-delay products and random loss,” IEEE/ACM Trans. Net- working,vol.5, pp. 336-350,June1997.Available:http://www.ece.ucsb.edu/Fac-ulty/ Madhow/Publications/ton97.ps

[41] M. Mathis, J. Semke, J. Mahdavi, and T. Ott, “The macroscopic behavior of the TCP congestion avoidance algorithm,” ACM Comput. Commun. Rev., vol. 27, no. 3, July 1997. Available: http://www.psc.edu/networking/papers/ model sub ccr97.ps

[42] C. Hollot, V. Misra, D. Towsley, andW.-B. Gong. (Apr. 2001). On designing improved controllers for AQM routers supporting TCP flows, in Proc. IEEE Infocom. Available: http://www-net.cs.umass.edu/papers/papers.html

[43] S.H. Low, F.Paganini, J.Wang, S.A. Adlakha, and J.C. Doyle, “Linear stability of TCP/RED and a scalable control,” in Proc. 39th Annual Allerton Conf. Communication, Control, and Computing, Monticello, IL, Oct. 2001. Available: http://netlab.caltech.edu

[44] S. Athuraliya and S.H. Low, “An empirical validation of a duality model of TCP and queue management algorithms,” in Proc. Winter Simulation Conf., Dec. 2001.

[45] Steven H. Low, Fernando Paganini, and John C. Doyle , “Internet Congestion Control” in IEEE Control Systems Magazine 2002

[46] JACOBSON, V. Congestion avoidance and control. In Proceedings of SIGCOMM ’88 (Stanford, CA, Aug. 1988), ACM.