An Improved Algorithm for Congestion Management in Network Based on Jitter and Time to Live Mechanisms

Samar Taha Yousif [1], Zaid Abass Fadahl Al-Haboobi[2]

Authors affiliations:
1*) Dept. of Computer Techniques Eng., Baghdad College of Economic Sciences University, Baghdad, Iraq
samaral_ani@baghdadcollege.edu.iq

2) Dept. of Computer Sciences, Baghdad College of Economic Sciences University, Baghdad, Iraq
zaid.alhaboobi@baghdadcollege.edu.iq

Paper History:

Received: 2nd June 2020

Revised: 17th July 2020

Accepted: 25th Sep. 2020

Abstract

As internet network developed rapidly in the past ten years, and its operating environment is constantly changing along with the development of computer and communication technology, the congestion problem has become more and more serious. Since TCP is the primary protocol for transport layers on the internet, the data transmitted via the transport protocol utilizes Vegas Transmission Control Protocol (TCP) as the congestion control algorithm, where it uses increasing in delay round trip time (RTT) as a signal of network congestion. However, this congestion control algorithm will attempt to fill network buffer, which causes an increase in (RTT) determined by Vegas, thereby reducing the congestion window, and making the transmission slower, Therefore Vegas has not been widely adopted on the Internet. In this paper, an improved algorithm called TCP Vegas-A is proposed consist of two parts: the first part is sending the congestion window used by the algorithm for congestion avoidance along with the TTL (Time To Live) mechanism that limits the lifetime of a packet in the network. While the second part of the algorithm is the priority-based packet sending strategy, and jitter is used as a congestion signal indication. The combination of the two is expected to improve the efficiency of congestion detection. A mathematical model is established, and the analysis of the model shows that the algorithm has better effects on controlling congestion and improving the network throughput, decreasing packet loss rate and increasing network utilization, the simulation is done using NS-2 network simulation platform environment and the results support the theoretical analysis.

Keywords: Congestion Control, TCP Vegas, TCP Vegas-A, Jitter, TTL, Throughput, Delay, Congestion Window

خوارزمية محسنة لإدارة الازدحام في الشبكة على أساس التزاحم والوقت إلى آليات حية

سمر طه يوسف ، زيد عباس فضل الحبوبي

الخلاصة:

مع تطور شبكة الإنترنت بسرعة خلال السنوات العشر الماضية ،و بالإضافة للتغير في بيئة تشغيله بشكل مستمر وسريع مع تطور تكنولوجيا الحاسبات والاتصالات ، أصبح من الضروري البحث عن حل لمشكلة الازدحام داخل الشبكة. نظرًا لأن بروتوكول TCP يستخدم في نقل البيانات على الإنترنت ، فإن البيانات المرسلة تستخدم (Vegas-TCP) كخوارزمية التحكم في الازدحام ، حيث تستخدم الزيادة في تأخير وقت الذهاب والعودة (RTT)كإشارة من ازدحام الشبكة. ومع ذلك ، ستحاول خوارزمية التحكم في الازدحام ملء المخزن المؤقت في الشبكة ، مما سيؤدي إلى زيادة (RTT) المحسوبة بواسطة Vegas ، و

يقلل من نافذة الازدحام ، ويجعل الإرسال أبطأ ، وبالتالي لم يتم اعتماد Vegas على نطاق واسع على الإنترنت.

في هذه الورقة ، تم اقتراح خوارزمية محسنة تسمى TCP Vegas-A تتكون من جزأين: الجزء الأول يرسل نافذة الازدحام التي تستخدمها الخوارزمية لتجنب الازدحام جنبًا إلى جنب مع آلية (Time to Live TTL (التي تحد من عمر الحزمة اثناء انتقالها داخل الشبكة. و الجزء الثاني من الخوارزمية هو استراتيجية إرسال الحزمة المستندة إلى الأولوية ، ويستخدم معها (jitter) كمؤشر لإشارة الازدحام. ومن المتوقع أن يؤدي الجمع بين هذين الجزئين إلى تحسين كفاءة الكشف عن الازدحام داخل الشبكة. تم إنشاء نموذج رياضي ، ويظهر تحليل النموذج أن الخوارزمية لها تأثيرات أفضل على التحكم في الازدحام ، تحسين إنتاجية الشبكة ، خفض معدل فقدان الحزمة و زيادة استخدام الشبكة . تمت المحاكاة باستخدام منصة NS-2 حيث دعمت نتائج التحليل النظري.

الكلمات المفتاحية: التحكم في الازدحام ، TCP Vegas، TCP Vegas-A، Jitter، TTL ، الإنتاجية ، التأخير ، نافذة الازدحام

1. Introduction

Due to the rapidly development of Internet in recent years, the number of Internet users has also grown rapidly where a great number of users are using excessive resources in a variety of network applications under limited resources and because the Internet is an open environment it cannot limit the number of clients according to the situation of resources. The users began to have some problems, among them, the occurrence of congestion which is a very important problem [1]. Congestion in the network can cause the nodes to drop a lot of data packets, which not only affects the quality of network services but also wastes the precious energy of the nodes, thus shortening the network life cycle. TCP [2] is currently the most commonly used protocol on transport layer in the Internet which includes a congestion control mechanism to guarantee the delivery of packets.

Researches have recently suggested some solutions to congestion in the network, Literature [3] refer to a congestion control method based on the rate of packet loss and delay to measure congestion. While the mechanisms proposed in the literature [4] divides network congestion into 5 levels based on delay jitter to distinguish between link loss, general congestion, and severe congestion. However, the reliability of the timing delay jitter classification has not verified yet. In literature [5] an analytic expression model for delay and jitter is proposed to detect traffic load, bandwidth and latency in the network, and this model were used in [6] as a metric to evaluate the optimal routing scheme for traffics which is sensible to delay or jitter. On the other hand, the authors in literature [7] made a comparison study between TCP Vegas and other TCP variants on the same network, and find that TCP Vegas achieves better throughput than others. According to [8], [9] TCP Vegas fails to be used on a large scale on the internet is mainly because of the use of TCP Vegas insufficient bandwidth competitiveness. While literature [10] proposed an improved multi-channel congestion control algorithm TCP Vegas based on packet queuing delay. This improved algorithm converts the congestion control problem of a single communication link into load balancing of multiple communication links the problem is that the data packets injected into a congested link can be shunted to other communication links, to achieve the purpose of improving the overall communication link bandwidth utilization.

On the other hand, in literature [11] authors used fluid-flow approximation to analyze the efficiency of the New Reno and Vegas TCP congestion control algorithm on the evolution of congestion window, the probability of packet-loss, and the duration of the queuing. Where TCP Vegas provided a fair service compared with Reno [11]. While in literature [12] authors focused on the efficiency of the actions between TCP Reno and TCP Vegas when they share the same router bottleneck. Where a drop tail and RED algorithm is used on the router and it has been found that the TCP Vegas does not have better drop-tail efficiency due to the difference in router buffer occupation. Authors in [13] show a comparison of the TCP variants used in network congestion control, They took into account the basis of different performance metrics include queue dimension, throughput, and packet delay rate. Within a low-cohesive network, Reno gives the best results than TCP Vegas.

For the problems of the above network congestion control and avoidance mechanism, this paper proposes an improved congestion avoidance algorithm for TCP Vegas, through sending data after sending the congestion window, and the priority-based number is used to send data in the current window according to the packet scheduling strategy, while jitter and TTL were used as a congestion indicator.

2. Materials and Methodology

An improved algorithm called TCP Vegas-A to avoid congestion in network nodes is proposed. The algorithm contains the congestion avoidance mechanism based on sending congestion window allocation and TTL to prevent local congestion. Table (1) cites the four stages of congestion control. While the important factors used by the algorithm are shown in Figure (1).

Table (1): The four phases of the congestion control algorithm

Phase No.	Description
phase-1	slow start algorithm
phase-2	congestion avoidance algorithm
phase-3	fast retransmission algorithm
phase-4	fast recovery algorithm

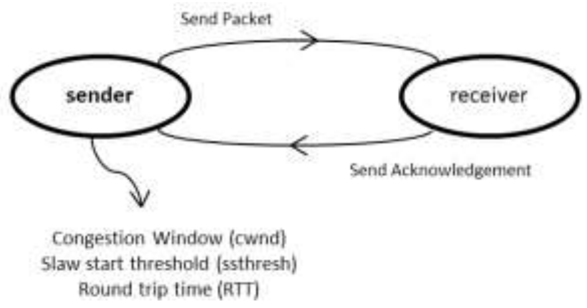

Figure (1): Common factors used in congestion control

The algorithm was proposed based on the survival period of the packet to shorten the delay on the network without causing great pressure on the reverse link. TTL which refers to the maximum age of the packet and represented by a byte in the header of the Internet layer, it will be tested at the beginning of each transmission to determine whether the packet will be forwarded or discard one by one. Each time a packet travels by a router, the TTL value decreases by 1. If the value of TTL reaches 0, the packet is dropped and an ICMP (Internet Control Message Protocol) packet is sent to the original sender. The algorithm uses this feature of the packet to classify the packet and process it in different ways.

As it's known that not all packets in a data stream have the same delay, and the delay of each packet varies with the status of the transmit network, if the network is not congested, there is no queue on the router and the total delay is composed of the line delay and propagation delay of each packet, Hence, network delay is the smallest at this time. If network congestion occurs, Queuing delay can affect end-to - end delay and cause changes in the delay of packets transmitted through the same link. The degree of change in the packet delay is called jitter. So, when a packet jitter become larger and larger, it can be considered that the network has potential congestion risks. The calculation of packet movement can use the following method:

For a specific data packet, we define the following parameters:
S (K) = the time when the sender generates the Kth packet.
R (K)= the time when the Kth packet is received by the receiver.
D (K) = the delay difference between packet K and packet K-1.

In this way, the end-to-end network delay of the Kth packet is R (K)-S (K).
From this we can get:

$$D(K) = (R(K) - S(K)) - (R(K-1) - S(K-1)) = (R(K) - R(K-1)) - (S(K) - S(K-1)). \quad ...(1)$$

If the network is not congested, then the value of D (K) from eq. (1) is 0; otherwise, D (K) is not 0 and gradually increase.

In a network that uses TCP Vegas-A as a congestion control algorithm, if congestion is detected on a routing node and the TTL of the arriving packet is relatively large, it means that the packet is relatively close to the source and is far from the receiver. At this time the delay jitter can be used as a warning sign of network congestion. So when the packet jitter is large, the network can be considered to be blocked. At this time, an ICMP - Source Quench message is generated (ISQ) to be sent to the source to inform the sender to adjust its sending rate in time, ISQ packets can be quickly forwarded to the source through few forwarding's so that it will not cause great pressure on the reverse link. When the sender receives the feedback of the receiver's loss event rate, it can calculate the rate and adjust the sending rate according to the calculated value. Also, delay jitter must be calculated. If there is no packet loss, determine whether the jitter is too large and adjust the sending rate. If the jitter is small, the size of the congestion window is increased by 1 for each returned acknowledgment packet: if the jitter is too large, the window size is 1 / cwnd for each acknowledgment packet received.

One of the tasks of the receiver is to allow the sender to calculate the RTT (Round Trip Delay time of the data packet) through feedback. When the connection starts the sender will use the RTT calculated for the first segment sent. And it will keep calculating RTT every pre-specified time (500ms) [14]. Another and most important task for the receiver is to calculate the loss removing event rate and feed it back to the sender, where it can calculate the rate and adjust the sending rate according to the calculated value. Also, it will calculate the packet delay. Each time the receiver makes feedback, it adds the sequence number of the most recently received message and the time the message was received. In this way, the sender can calculate the RTT time.

On the other hand, if the TTL is relatively small, it means that the packet has been transmitted for a long time, the algorithm assumes that the packet will soon reach the receiver, then a throughput (TH) parameter is calculated form eq. (2), when the TTL is greater than TH, the algorithm use ICMP - Source Quench message to warn the sender of the situation of congestion within the network.

For a single connection, according to the formula given by S. Floyd in [15], the maximum throughput is:

$$TH \leq \frac{1.5\sqrt{2/3}*B}{R*\sqrt{p}} \quad ... (2)$$

Where, B is the packet size (the default is 512 byte), R is an RTT, and p is the average packet loss rate.

The simulation is realized with the NS-2 network simulation platform. The experimental topology is shown in figure (2). Where TCP proxy is running on source 1 (S1), the data source is P, and it is running on destination 1 (D1). After running the TCP link, it returns an ACK confirmation packet after receiving the TCP packet. Figure (2) shows the simulation network, while table (2) shows simulation parameters.

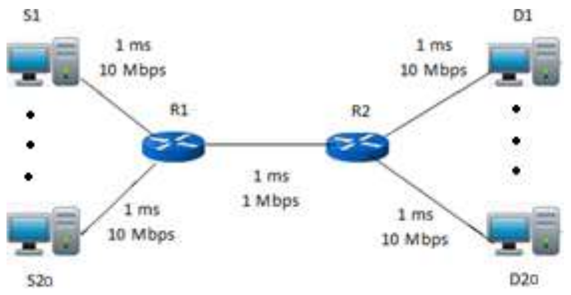

Figure (2): Simulation Network

Table 2 Simulation Parameters

Parameter	Value
TCP Protocol	TCP Vegas and TCP Vegas-A
TCP Packet Size	1000 byte
Access Link Delay	1 ms
link rate between nodes	10 Mbps
link rate between R1 and R2	1 Mbps
Maximum Congestion Window Size	300
Simulation Time	60s

3. Results and Discussion

The experimental results show that the average throughput rate of TCP Vegas is 148.32 packets/second, and the average throughput rate of TCP Vegas-A is 161.78 packets/second, means that Vegas-A has better TCP friendliness. The average packet delay of TCP Vegas is 2.3920 (100ms), while it's 1.3434 (100ms) in TCP Vegas-A, this showing that the queuing delay in TCP Vegas-A is not increased much. Calculating packet jitter at the receiving end of TCP Vegas and TCP Vegas-A, the jitter of TCP Vegas is 0.0966 ms, while the jitter of TCP Vegas-A is 0.0152 ms which is about 8% lower than the jitter of TCP Vegas, it can be seen that TCP Vegas-A is more suitable for data transmission. A comparison between Vegas and Vegas-A is shown in Table (3).

Table (3): Comparison of two algorithms

Algorithm	Average Throughput (packet/s)	Average Packet Delay (100ms)	Average Packet Jitter (ms)
TCP Vegas-A	161.78	1.3434	0.0152
TCP Vegas	148.32	2.3920	0.0966

It can be seen from the table (3) that TCP Vegas-A is higher in throughput than TCP Vegas, it is about 13.46 packets / s, and it is about 1(100 ms) and 0.08 ms lower in packet delay and packet jitter respectively than TCP Vegas. This shows that under the condition of guaranteeing certain service quality, TCP Vegas-A is effectively controlled the network congestion, also takes into account the friendliness and fairness of TCP services.

4. Conclusions

This paper proposes a new congestion avoidance algorithm called TCP Vegs-A. The algorithm is analyzed by a simplified mathematical model and the feasible basis of the algorithm is given. The algorithm implemented in NS-2 network simulation platform, and the expected results are achieved, the simulation results show that through the estimation of jitter and TTL improves the effective throughput, reduces the average delay of the network and effectively avoiding the network packet loss caused by the overflow of the flush area. The detection of potential congestion hazards in the network further improves the efficiency of network transmission.

5. References

[1] C. Systematics, "effective practices for congestion m anagement: final report," vol. 24, no. November, 2008.

[2] P. Akiene and L. Kabari, "Simulation of an Optimized Data Packet Transmission in a Congested Network," Netw. Complex Syst., vol. 5, no. 8, pp. 29–37, 2015.

[3] Z. Wang, X. Zeng, X. Liu, L. Chen, and B. Guo, "TCP congestion control algorithm for heterogeneous networks," Dianzi Yu Xinxi Xuebao/Journal Electron. Inf. Technol., vol. 38, no. 4, pp. 780–786, 2016, doi: 10.11999/JEIT150774.

[4] N. Ding, R. Q. Wu, and H. Jie, "TCP BRJ: Enhanced TCP congestion control based on bandwidth estimation and RTT jitter for heterogeneous networks," Lect. Notes Electr. Eng., vol. 322, pp. 623–632, 2015, doi: 10.1007/978-3-319-08991-1_64.

[5] H. Dahmouni, A. Girard, and B. Sansò, "An analytical model for jitter in IP networks," Ann. des Telecommun. Telecommun., vol. 67, no. 1–2, pp. 81–90, 2012, doi: 10.1007/s12243-011-0254-y.

[6] H. Dahmouni, A. Girard, M. Ouzineb, and B. Sansò, "The impact of jitter on traffic flow optimization in communication networks," IEEE Trans. Netw. Serv. Manag., vol. 9, no. 3, pp. 279–292, 2012, doi: 10.1109/TNSM.2012.051712.110148.

[7] B. S. Yew, B. L. Ong, and R. B. Ahmad, "Performance evaluation of TCP vegas versus different TCP variants in homogeneous and heterogeneous wired networks," World Acad. Sci. Eng. Technol., vol. 50, no. February, pp. 175–181, 2011.

[8] Y. Cao, M. Xu, and X. Fu, "Delay-based congestion control for multipath TCP," Proc. - Int. Conf. Netw. Protoc. ICNP, no. January 2015,

2012, doi: 10.1109/ICNP.2012.6459978.

[9] B. A. A. Nunes, K. Veenstra, W. Ballenthin, S. Lukin, and K. Obraczka, "A machine learning framework for TCP round-trip time estimation," Eurasip J. Wirel. Commun. Netw., vol. 2014, pp. 1–22, 2014, doi: 10.1186/1687-1499-2014-47.

[10] E. Altman, C. Barakat, E. Laborde, P. Brown, and D. Collange, "Fairness analysis of TCP/IP," Proc. IEEE Conf. Decis. Control, vol. 1, no. May 2014, pp. 61–66, 2000, doi: 10.1109/CDC.2000.912733.

[11] A. Doma, J. Doma, M. Pagano, and T. Czach, "The Fluid Flow Approximation of the TCP Vegas and Reno Congestion Control Mechanism," pp. 193–200, In: Czachórski T., Gelenbe E., Grochla K., Lent R. (eds) Computer and Information Sciences. ISCIS 2016. Communications in Computer and Information Science, vol 659. Springer, Cham. doi: 10.1007/978-3-319-47217-1.

[12] T. Chowdhury and M. J. Alam, "Performance Evaluation of TCP Vegas over TCP Reno and TCP NewReno over TCP Reno," JOIV Int. J. Informatics Vis., vol. 3, no. 3, 2019, doi: 10.30630/joiv.3.3.270.

[13] P. Chaudhary and S. Kumar, "A Review of Comparative Analysis of TCP Variants for Congestion Control in Network," Int. J. Comput. Appl., vol. 160, no. 8, pp. 28–34, 2017, doi: 10.5120/ijca2017913087.

[14] G. Hasegawa and M. Murata, "Survey on fairness issues in TCP congestion control mechanisms," IEICE Trans. Commun., vol. E84-B, no. 6, pp. 1461–1472, 2001.

[15] S. Floyd and K. Fall, "Promoting the use of end-to-end congestion control in the Internet," IEEE/ACM Trans. Netw., vol. 7, no. 4, pp. 458–472, 1999, doi: 10.1109/90.793002.