MODIFIED VIRTUAL CLOCK SCHEDULING ALGORITHM FOR QUEUE MANAGEMENT

P. SANKAR[1], C. CHELLAMUTHU[2]

Queue management is an important issue in today's high-speed packet networks which supports a variety of applications with different service necessities. The Virtual Clock (VC) algorithm provides flow with a certain throughput and a small queuing delay. However, in the presence of video traffic, it is important to ensure that co-existent responsive flows (TCP) are not starved. In this paper, a modification to VC algorithm has been proposed to ensure fairness to responsive flows in presence of nonresponsive flows (UDP), while retaining the advantages of VC algorithm

Keywords: Video, NS2, Virtual Clock, MVC, Scheduling

1. Introduction

One of the most noteworthy promises in today's Internet is to offer Quality of Service (QoS) guarantees. The User Datagram Protocol (UDP) and Transmission Control Protocol (TCP) packets coexist in the Internet. Congestion can arise due to the non responsive nature of UDP based flows. The UDP based flows are potentially troublesome because they escalate the packet loss rates in the network and can ultimately cause congestion collapse [1] [2]. Inside the routers, TCP and UDP packets need to coexist occupying the buffer space. Numerous works have been carried out to offer routers with mechanisms for shielding responsive flows against non responsive flows [3] [4].

The packet scheduling algorithms can impart guaranteed QoS in network routers and switches. These algorithms not only provide a protection between the streams but also allow packets from various traffic streams to be multiplexed. In case of an overflow in switch buffers packets get discarded. The main task of packet scheduling disciplines is to determine which packet among different service classes gets transmitted. These operations have an effect on parameters such as throughput, delay, and loss rate. The packet scheduling mechanisms are classified into two categories: work-conserving and non-work-conserving [5]. When there are no packets waiting in the router's queues the work-conserving

[1] Jerusalem College Of Engineering, ANNAUNIVERSITY, India, e-mail: sankarp1@ieee.org
[2] RMK Engineering College, ANNAUNIVERSITY, India, e-mail: ccm.eee@rmkec.ac.in

scheduler is idle. In a non-work-conserving scheduler such as VC, each packet is allocated a time when it has to be dispatched to the output interface. The scheduler remains at rest until the time when the next packet is eligible to be sent.

When choosing the next packet to be dequeued, VC shortens the calculation of finish times used by the scheduler [6] [7]. This algorithm also provides for each, link bandwidth allocation and safeguard between traffic flows. A method has been projected for effective realization of fairness between nonresponsive and TCP flows, while maintaining a steady queue [8]. A study on the queue management algorithms using optical network (OPNET) simulation tool has been made by Chandra et al [9]. A new network queue management algorithm, CCU (Compare and Control Unresponsive flows) was planned in order to fortify the heftiness of Internet against nonresponsive flows [10]. Apart from nonresponsive UDP traffic, hostile TCP flow pretense a serious confronts to congestion control and stability of the Internet [11]. The Fair Early Drop (FED) scheme prevents the unfairness problem by making sure that the flows do not devour more than their reasonable share of network resources. It is taken care of by dropping more packets from the flows [12].The performance of Adaptive Active queue management (AAQM) algorithm in the company of feedback delays and its capability to maintain a very small queue length and sturdiness in the presence of traffic burst has been studied [13]. A set of router-based QoS mechanisms which includes queue strategy, resource provisioning, and flow metering has been proposed [14].These schemes provide rate assurance to UDP flows, safeguard the well behaved TCP flows against non responsive flows and bandwidth fairness among the flows. The present traffic flow model in the Internet indicates that considerable percentage of volume is video/UDP traffic. The authors have studied in detail the effect of video traffic on responsive flows using queue management algorithms [15]. A Weighted queue WRIO algorithm to realize relative discrimination of different drop precedence and to improve link utilization, for the differentiated networks with TCP traffics has been discussed [16]. An effective-adaptive virtual queue (AVQ) algorithm, to increase the responsiveness and toughness of the transmission control protocol has been proposed [17]. The authors have developed a simulation environment with network simulator (NS2), ffmpeg codec and its service utilities to learn the performance of video traffic under various scheduling algorithms [18]. It is seen that the VC algorithm gives each link bandwidth allocation, protection among traffic flows and improved throughput. But it has no method to protect the well behaved TCP flows against non responsive flows. In this paper a modification to the VC algorithm is proposed to achieve fairness for responsive flows in the presence of non responsive flows such as video, while retaining the advantages of the VC algorithm.

This paper is organized as follows. In Section 2 a modification to the VC algorithm to achieve fairness has been discussed. A simulation environment using ns2 and its utilities are discussed in Section 3. The results of simulation for a typical network with various traffic sources, employing VC and MVC algorithms are presented in Section 4.The conclusion and discussion are in Section 5.

2. Modified Virtual Clock algorithm (MVC)

The MVC is a queue management algorithm evolved from VC algorithm. VC does not provide fairness among TCP and UDP flows. UDP is a non responsive traffic protocol, while TCP is responsive. During congestion TCP stops transmission but UDP continues to transmit. It doesn't allow TCP to flow under congestion. Hence in VC, TCP doesn't get a reasonable share of the network resources. To overcome this disadvantage reservation is made for the data flows in MVC. Reservation for a single flow makes the scheduler unfair. Hence reservation is done for both TCP and UDP.

Working of MVC

The MVC consists of three stages. The first stage consists of classification of TCP and UDP packets for providing reservation. The last two stages, data forwarding and flow monitoring are the same as in VC algorithm.

Filtering and Reservation

In the first stage of MVC algorithm, the packets from TCP and UDP flows are classified. The headers from the packets are identified. This header is used to obtain parameters like source address, destination address, source and destination port number and header size. These parameters are used to classify packet as TCP or UDP packets. After the classification of TCP and UDP packets, the thresholds are updated. There are two thresholds T1 and T2 used for assigning reservation for TCP and UDP. When simultaneous flow of UDP and TCP packets exist, then reservation is increased to allocate the resources fairly. When either TCP or UDP flows exists, the flow is simple and doesn't require any reservations. This is achieved by adaptively updating threshold T1. Each time the TCP or UDP counter reaches 0, T1 is reduced. When there are seven continuous incoming UDP packets or three continuous UDP packet drops or six continuous incoming TCP packets or two continuous TCP packet drops, T1 is increased.

When there is no TCP packets flow and only UDP flows exist, the queue reserved for UDP is increased. When there is no UDP packets flow and only TCP flows exist, the queue reserved for TCP is increased. So the thresholds are updated adaptively upon the packet flow. This is achieved by adaptively updating threshold T2. When there are seven continuous incoming UDP packets or three continuous UDP packet drops, then the threshold T2 is increased to utilize the region reserved for TCP packets. To provide reservation for TCP flows, the

threshold T2 is decreased adaptively when seven continuous incoming TCP packets occur.

The queue schematic is shown in Fig 1. The T1 is the length of common queue. The length (1-T1) T2 is reserved for UDP whereas (1-T1) (1-T2) is used for TCP. The threshold T1 is varied between 0.70 to 0.90 and threshold T2 is varied between 0.0 to 1.0. These values decide the degree of convergence/adaptation of the threshold.

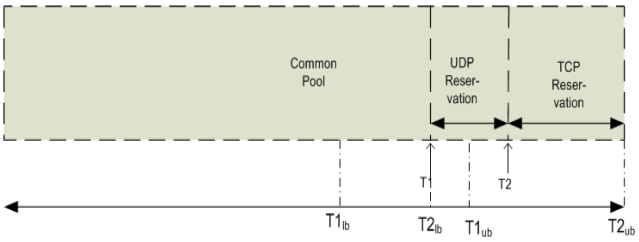

Fig.1 MVC Queue Schematic Diagram

After updating the thresholds, the packets are en-queued. The queue utilization is calculated to decide whether reservation should be done or not. If the queue utilization is below threshold T1, both TCP and UDP packets are treated uniformly. When it exceeds T1, and if the incoming packet is TCP, the packet is enqueued without affecting UDP reservation because the TCP packet is enqueued only when vacancy exists in TCP reservation. If the incoming packet is UDP, the packet is enqueued without affecting TCP reservation because the UDP packet is enqueued only when vacancy exists in UDP reservation.

Data Forwarding

Data forwarding is used to enqueue the packets into the outgoing queue. Upon receiving the first packet from the flow, both the VC and auxVC are equated to real time. When the subsequent packets are received from the flow, the VC and auxVC are advanced with an increment of Vtick. If a packet is received late from the flow, then auxVC starts and packet is stamped with the current time. The subsequent packets are stamped with an increment of Vtick to the previous value. If all the packets have constant size then Vtick=1/ARi (packet/size). Then the packets are inserted into outgoing queue and served in the order of escalating stamp values.

Flow monitoring

Average interval rate is the maximum number of packets a flow can transmit. It can be computed as AIRi = ARi *AIi at flowi setup. Upon receiving every set of AIRi data packets from flowi, the subsequent conditions are checked. In the first condition when the difference between VC and real time is greater than threshold value ‘T’ (as mentioned in the VC algorithm); a warning message is sent to flow source. If the Virtual Clock is lesser than real time (i.e.) if the flow

has transmitted lesser number of packets, then VC value is equated to real time. This is done to prevent the flow from transmitting the number of packets that it missed to transmit in the last interval.

Proposed Modifications

The steps followed in the MVC algorithm are as follows.

Header is identified from the packet to get the header size for classification.

Threshold T1 is updated.

T1 is increased, if simultaneous TCP/UDP flows exist.

T1 = T1 + 0.1

If(T1>0.9)

T1 = 0.9

T1 is decreased, if either TCP or UDP flows exist.

T1 = T1 – 0.1

if(T1<0.7)

T1 = 0.7

Threshold T2 is updated.

If UDP packet, T2 is increased.

If(T2>1.0)

T2 = 1.0.

If TCP packet, T2 is decreased.

If(T2<0.0)

T2 = 0.0

Queue usage is calculated.

If usage is lesser than threshold T1, packets is forwarded to enqueue stage.

Else

If TCP and vacancy in TCP resv exist, packet is forwarded to en-queue stage; else drop packet.

If UDP and vacancy in UDP resv exist, packet is forwarded to en-queue stage; else drop packet.

Packet enters the Virtual Clock stage.

Increment byte count.

If byte count > bytes per interval, then

Byte count = byte count – bytes per interval.

If Virtual Clock > current time

Source is sending fast.

Else if Virtual Clock > current time

Source is sending slowly.

AuxVC = max(current time, AuxVC

Increment Virtual Clock and AuxVC.

Packet is time stamped with AuxVC and enqueued into the queue in the ascending order of time stamp.

If queue limit exceeded

Drop the packet at the queue end.

3. Simulation Environment

The video streaming environment developed consists of NS2, ffmpeg codec and its service utilities [19]. The NS2 is an object oriented network simulator [21][22]. It ropes the simulation of TCP, UDP, media access control layer protocols and various routing and multicast protocols over both wired and wireless networks. There are four schedulers available in the network simulator: linked list, heap, calendar queue and real time scheduler. The scheduler initiates group of events one after other and runs it into completion. The software tool ffmpeg in the MPEG4 format is chosen as video codec for encoding the test video because it is widely used in the real world and provides better documented methods for comparing video qualities. The service utilities of Video codec consists of error generator (eg), etmp4, psnr, hist and mos. The NS2 simulator has several tools such as network animator (nam), xgraph and gnu plot. These utilities help in generating and visualizing the simulation results.

A standard dumbbell topology is considered for simulation of the network. The simulation model of the network is shown in Fig 2.The network comprises of 5 sources, two routers and 5 destinations. Two flows are considered as test flows for the analysis of the performance of the algorithms. Responsive flow is a constant bit rate packet which flows from node 1 to destination 1 and the non responsive flow is a video which flows from node 5 to destination 5. The parameters used for simulation is given in Table1. To characterize the network behavior, three QoS metrics throughput, latency, and end-to-end delay jitter are considered. These parameters along with the results are explained as follows.

Table 1

Simulation Parameters

Network Simulator	NS2
Algorithms	VC,MVC
Simulation time	9sec
No of nodes	10
Data Packet size	1024,1000
Routers	Two
Sender	Five
Receiver	Five

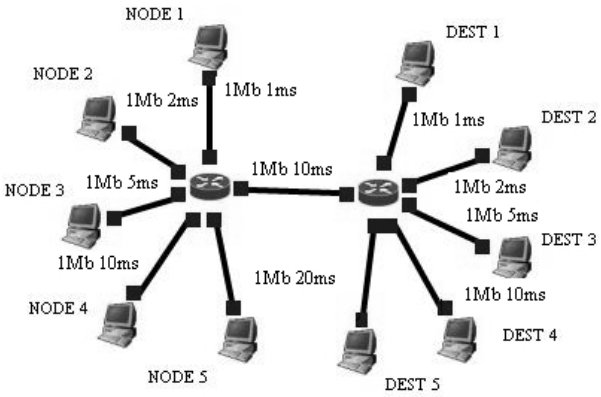

Fig. 2 Network topology

4. Results

The results and analysis of the above network are described as follows.

Throughput

Throughput is expressed as the ratio of the total no of receiving packets to total no of packets sent. The variation of throughput for VC and MVC algorithms for different scenarios of responsive and nonresponsive flows are shown in Fig 3 and Fig 4 respectively. The scenarios are denoted by the term 2u8t, 4u6t etc.

The term 2u8t means 2 nonresponsive flows and 8 responsive flows as shown in Fig 3.This has been further increased in such a way that nonresponsive sources exceed the responsive sources. When the flow is responsive, the MVC algorithm performs well to give a better throughput as compared to the VC algorithm. In the case of non responsive flows the MVC algorithm initially gives a slightly lesser throughput as shown in Fig 4. But as the number of non responsive flows goes on increasing the modified algorithm gives a better throughput than the VC algorithm. There is a fluctuation in the throughput due to the delay experienced in the links.

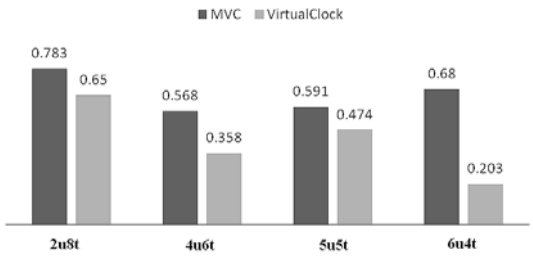

Fig. 3 Packets received for Responsive Flow

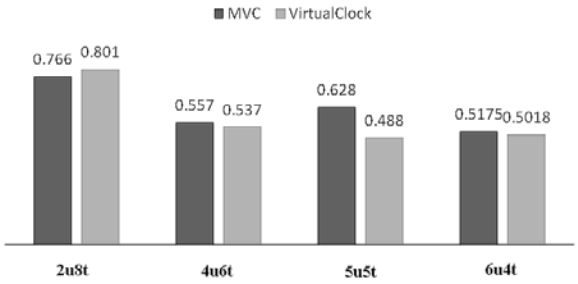

Fig. 4. Packets received for Nonresponsive Flow

Latency

The term latency refers to end to end delay incurred in the transmission of network data. The peak bandwidth of a network connection is fixed according to the technology used. The actual bandwidth that is obtained varies over time and is affected by high latencies. Excessive latency creates bottlenecks that prevent the data from filling the network, thus decreasing effective bandwidth. The impact of latency on network bandwidth can be temporary (lasting a few seconds) or persistent (constant) depending on the source of the delays.

The end to end delays for responsive and nonresponsive flows are shown in Fig 5 and 6 respectively. The MVC algorithm gives lesser amount of queuing delays when compared to VC algorithm.

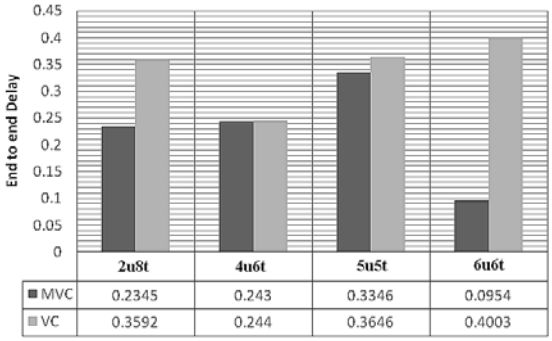

Fig. 5 Latency experienced by Responsive Flow

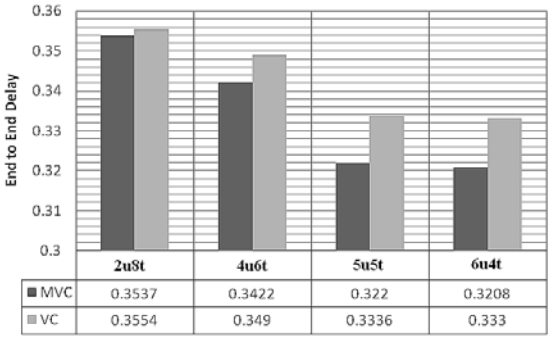

Fig 6. Latency experienced by Nonresponsive Flow

Delay variations

The definition of delay jitter, which is based on the end-to-end delay, is defined as the absolute difference between the delays of two consecutive packets. Assume Xi is the end-to end delay of the i^{th} packet, and then the end-to-end jitter is given as:

$$\text{Jitter} = \left| Xi+1 - Xi \right| \quad (1)$$

For audio/video applications, delay jitter is one of the key parameter that affects the QoS.

The average delay jitter for different scenarios is calculated and shown in Table 2. This clearly indicates that proposed MVC gives a smaller delay jitter than VC algorithm for all the scenarios.

Table 2

Average delay jitter

Scenarios/Algorithms	Virtual Clock	Modified Virtual Clock
2u4t	0.001272	0.001257
4u6t	0.001299	0.001299
5u5t	0.013749	0.001067
6u6t	0.099274	0.001143

Weighted parametric Index Analysis

The WPI is used to study the performance based on cumulative gross metric.

Weight parametric index (WPI) is given by

$$\text{WPI} = \sum_{i=1}^{3} W_i * X_i \tag{2}$$

where,

Wi is the Assigned weight

Xi is the Relative index

The performance of nonresponsive flow obtained with MVC and VC algorithms is shown in Table 3

Table 3

Performance comparison based on parameter

Algorithms/Parameters	Packet Delivery	Latency	Average Jitter
MVC	A	B	A
VC	A	B	C

Throughput A= > 0.50, B = 0.30 to 0.50, C = 0.10 to 0.30

Latency A = < 0.30ms, B = 0.30 to 0.50ms, C = > 0.50ms

Average Jitter A= < 0.005ms, B = 0.005 to 0.012 ms, C= > 0.012 ms.

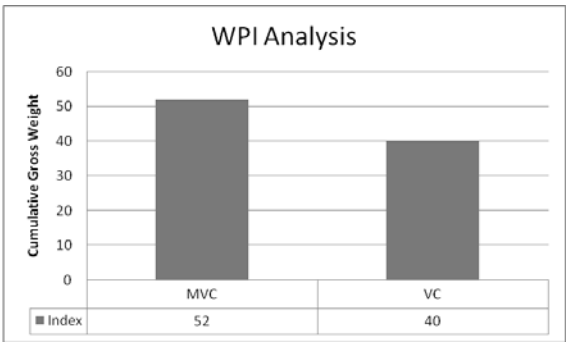

Fig. 7 Weighted parametric Index

The WPI has been calculated by assigning numerical weights to each of the parameters based on the rankings of the relative values. It is plotted as shown in Fig7. The analysis indicates that higher amount of fairness has been achieved by MVC compared to VC algorithm.

5. Conclusion

This paper describes the modifications proposed on VC scheduling algorithm and is named as MVC. The key idea is to incorporate fairness to responsive flows in the company of non responsive flows, at the same time retaining the advantages of VC algorithm

The comparative study of VC with MVC algorithm shows that, MVC provides 8% improvement in the overall performance index of the network for the specified throughput. The MVC algorithm gives increased throughput without compromising fairness even when non responsive flows are increased considerably. Further works are in progress to test this algorithm by fusing it in a real router.

R E F E R E N C E S

[1]. *V. Jacobson.* Congestion Avoidance and Control, IEEE/ACM Transactions on Networking, (TON),**vol 18**,no.4,pp.1275-1288, August 1988.

[2]. *K. Fall and S. Floyd,* Router Mechanisms to Support End-to-End Congestion Control, IEEE/ACM Transaction on Networking, **vol 8**, no.5 February 1998.

[3]. *V. Paxson* End-to-End Internet Packet Dynamics. IEEE/ACM transactions on Networking, **vol 7**, no.3, June 1999.

[4]. *D. Lin and R. Morris.* Dynamics of Random Early Detection, Proceedings of ACM SIGCOMM, Computer Communication Review, **vol 27**, no 4.pp.127-137, September 1997.

[5]. *J.C.Saez, J.I.Gomez, M.Prieto,* Improving Priority Enforcement via Non-Work-Conserving Scheduling, Proceedings of ICPP '08. 37th International Conference on Parallel Processing, pp 99 - 106, Portland, 9-12 Sept 2008.

[6]. *Lixia.Zhang,* Virtual Clock: A New Traffic Control Algorithm for Packet-Switched Networks, IEEE paper, **vol 9**, No.2, pp 101-124, 1991.

[7]. *Lain-Chyr Hwang, San-Yuan Wang Steen J. Hsu, and Yong-Hua Huang* "A New Scheduling Algorithm: Jumping Virtual Clock, Proceedings of 19th International Conference on Advanced Information Networking and Applications (AINA'05) **vol 2**, Taipei, Taiwan, March 25-30, 2005.

[8]. *Jong-hwan Kim, Hyunsoo Yoon, Ikjun Yeom*, Active Queue Management for Flow Fairness and Queue Stability, IEEE Transactions on Parallel and Distributed Systems, **vol 13**, pp. 998-1000, May. 2010.

[9] *H.Chandra, A. Agarwal and T .Velmurugan*, A number of active queue management algorithms for TCP/IP network using OPNET Simulation Tool, International Journal of Computer Applications, **vol 6**, No11, pp.12 -15, 2010.

[10]. *Liyuan Zhao, Keqin Liu and Jun Zheng*,A New AQM Algorithm for Enhancing Internet Capability Against Unresponsive Flows,Computational Intelligence and Security, Springer, **vol 3802/2005**, pp.711-718, 2005

[11].*Sungwon Yi, Xidong Deng, George Kesidis and Chita R. Das*,A dynamic quarantine scheme for controlling unresponsive TCP sessions,Telecommunication Systems,Springer,**vol 37**, Number 4, pp 169-189,2008.

[12]. *G. Aldabbagh, M. Rio, I. Darwazeh*, Fair Early Drop: An Active Queue Management Scheme for the Control of Unresponsive Flows", Computer and Information Technology (CIT), 2010 IEEE 10th International Conference, pp.2668 – 2675, 2010.

[13]. *Rahim Rahmani, Christer Åhlund, Theo Kanter* Design of Active Queue Management for Robust Control on Access Router for Heterogeneous Networks", EURASIP Journal on Wireless Communications and Networking, **vol 2011**,2011

[14]. *S. H. Jeong, H. Owen, J. Copeland,J. Sokol,* QoS support for UDP/TCP based networks Computer Communications, **vol 24**, Issue 1,pp 64-77,2001.

[15]. *P. Sankar, C. Chellamuthu*, Study on the Effect of Video Traffic on Responsive Flows Using Queue Management Algorithms", Asian Journal of Information Technology, **vol 10**, pp. 42-46, 2011.

[16]. *Yang Xiaoping, Chen Hong, Zhang Zhenyu*, A Queue Management Algorithm for Differentiated Services, Intelligent Computation Technology and Automation (ICICTA), IEEE International Conference held at Shenzhen, Guangdong, China, pp.941 – 944, 2011.

[17]. *H. Wang, C.Liao, and Z.Tian*, Effective adaptive virtual queue: a stabilizing active queue management algorithm for improving responsiveness and robustness" IET Communication, **vol 5**, Issue 1, p.99–109, 2011.

[18]. *P. Sankar, C. Chellamuthu*, Study and simulation of Virtual clock Scheduling Algorithm for Video Streaming", International journal of Computer Theory and Engineering, **Vol1**, No. 5, pp 509-524, Dec 2009

[19]. *P. Sankar, C. Chellamuthu*, Study and simulation of Video streaming, International journal of Computers and Applications, ACTA press, **vol 32**, No 4, 2010**.**

[20]. *Kevin Fall and Kannan Varadhan*, 2005 the NS Manual.

[21]. http://www.isi.edu/nsnam/ns/ns2.