Research on multi-virtual queue based on flow estimation

Wenqing Huang[1, 2*]

[1]*Department of Computer and Information Engineering, Heze University, Heze 274015, Shandong, China*

[2]*Key Laboratory of computer Information Processing, Heze University, Heze 274015, Shandong, China*

Received 1 August 2014, www.cmnt.lv

Abstract

To solve the equity issues in the network bandwidth competition of TCP/UDP mixed flow, we have raised an AQM algorithm framework based on "granularity", which is aimed to realize the divide-and-rule goal of the conservative TCP and the greedy UDP so as to alleviate the equity issues of the network resources caused by different protocols by designing a common framework. Furthermore, under the AQM framework of "granularity", we have brought forth Multi-Virtual Queue based on Flow Estimation (VFQ) and we have also given the detailed implementation schemes for different modules of VFQ algorithm. In order to further verify the feasibility and validity of VFQ algorithm, we have designed 5 computer network congestion scenarios in AQM network simulation platform and made simulation comparisons with the classical ARED, PI and Blue algorithms. Numerous simulation results demonstrate that in contrast with other AQM algorithms, VFQ can better adapt to various sudden network congestion scenarios and it has better response speed and queue management performance.

Keywords: multi-virtual queue, flow estimation, TCP/UDP mixed flow

1 Introduction

Network burstiness is the main factor which causes the instable performances of AQM algorithm. However, most of the current AQM algorithms handle the network traffic based on "packet granularity". In face of the network with the coexistence of TCP and UDP flows, these algorithms can't guarantee the equity of the bandwidth allocation. In addition, since UDP flow is an open-loop non-response flow, it ignores ECN congestion feedback signal of AQM algorithm, making it impossible for AQM algorithm to adjust the resource occupancy of UDP flow through the feedback signal; therefore, the traditional AQM fails to control the tip nodes by only ECN feedback mechanism [1].

In order to solve these problems, this paper has first designed an AQM algorithm framework based on "granularity", based on which, it has raised Multi-Virtual Queue based Flow Estimation (VFQ). What this algorithm is different from the traditional AQM algorithms include:

I. VFQ introduces the idea of "granularity" and differentiates and maintains the traffic information passing through every bottleneck route;

II. In accordance with the different characteristics of UDP and TCP flows, it has constructed 2 logically-isolated virtual queues (VQ) respectively and designed different congestion control strategies for these 2 VQ;

III. Through the network load Perceive information of TCP/UDP flow, the system will adjust and control the relative sizes of these two VQ automatically.

2 Network congestion control mechanisms and active queue management (AQM)

When the demand of users for network resource (including CPU processing capability, the size of memory space and information channel transmission rate) exceeds the intrinsic network resource amount, the network will be in the state of "overloading" continuously. Under such circumstance, network congestion will occur, as shown in Equation (1). As for the computer network system, the occurrence of congestion exists objectively.

$$\sum \textit{the demand of all users for network resource} > \textit{network resource} . \quad (1)$$

2.1 ACTIVE QUEUE MANAGEMENT (AQM) MECHANISM

The source end network congestion control measure based on TCP protocol is essential and indispensable. However, there are certain limitations for such network congestion control measure. This measure is not always effective in successfully achieving congestion control for all networks [2]. Thus, it is necessary to add other network congestion control mechanisms to the central router so as to reach satisfactory effectiveness in either preventing network congestion in advance or remove network congestion after it occurs. In order to avoid the disadvantages of TCP source end network congestion control measure, IETF (Internet Engineering Task Force) firstly put forward to

[*]***Corresponding author*** email: Huangwenqing8686@126.com

Active Queue Management (AQM). AQM mechanism has the following three advantages:

1) Low loss in router node network packets: the transmission of data packets in computer network is featured with strong abruptness. When the free buffer memory space approaches to "0" or is just equal to "0", the router will fail to allocate certain buffer memory space for those abrupt network packets. When just a small part of the buffer memory space of a router is occupied, the router will have enough space to store those abrupt packets without needing to discard these packets. If AQM mechanism is not adopted, many packets will be discarded when the buffer memory space of a router is full. Such atmosphere is not good for the operation of computer network [3]. Three major reasons are: firstly, drop tail rule will make network to generate "overall synchronization", which may lead to users cannot fully utilize information channel bandwidth and meanwhile the network throughput will also decrease. Secondly, compared with the drop tail methods of simply discarding network packets, it is difficult for TCP to recover from the state of network congestion. Thirdly, meaninglessly discarding the packets is an expression of the fact transmission links cannot be effectively utilized [4].

2) Shortening the delay time of network service: if the instant queue size is made to fluctuate between small ranges. AQM mechanism will reduce the delay time of packet processing, which will make the instant data service of some internet better.

3) Removing "deadlock" behaviour: if AQM mechanism can always provide a memory space of a certain space to accept instant packet, it can avoid the occurrence of "deadlock" atmosphere.

2.2 SEVERAL TYPICAL AQM ALGORITHMS

2.2.1 ARED algorithms

The basic concept of ARED algorithm is to perceive whether RED should be aggressive or conserved through checking average changes of queue. It puts forward to an automatic regulation mechanism and set packet loss probability according to the changes of flow amount on the bottleneck link. Larger packet loss probability is suitable for more cases; vice versa. If the queue size is near *minth*, it indicates that the congestion control is too aggressive and the packet loss probability *maxp* should be reduced. The reducing degree is signified by "a", hence, $maxp=a\cdot maxp$; If the queue size is near *maxth*, it indicates that the congestion control is too conserved and the packet loss probability *maxp* should be increased. The increasing degree is signified by "b", hence, $maxp=b\cdot maxp$, where $0<a<1<b$.

2.2.2 PI algorithm

PI algorithm is a kind of AQM algorithm based on typical control theory. This algorithm effectively improves the performance of network system and can be applicable for the changes of network flow. PI controller simply considers the network as a linear time invariant system and takes the packet loss probability as the measurement of current network congestion. The discrete version of PI control algorithm is:

$$p(k) = p(k-1) + a(q(k) - q_{ref}) - b(q(k-1) - q_{ref}), \qquad (2)$$

where, *p(k)* refers to the packet loss probability at the time *k*; *a* and *b* respectively refers to the current queue size and the control parameter of the difference between the queue size at the previous moment and expected queue size; q_{ref} is the expected queue size. The algorithm control block diagram is as shown in Figure 1:

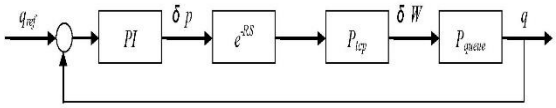

FIGURE 1 Systematic block diagram of PI algorithm

Although PI controller algorithm is simple and easy to achieve, its response time is relatively long. This disadvantage can be improved by introducing differentiation element to accelerate the response speed of the system [5].

2.2.3 Blue algorithm

Blue algorithm manages the queue by using queue overflow and the event of idle link, which is improvement on RED algorithm. It maintains a probability Pm to mark the drop probability. Pm will be increased if the event of packet loss happens caused by queue overflow in order to pick up the speed of sending congestion feedback information to the transmit end; on the contrary, Pm will be decreased if the queue is idle caused by idle link in order to slow down the speed of sending congestion feedback information to the transmit end. So the Blue algorithm can effectively control the speed of sending congestion feedback information [6].

Blue algorithm implements the congestion control according to the packet loss and link service condition directly. Compared with RED algorithm, it decreases packet loss and the demand for buffer memory space of the router, but there are still disadvantages for Blue algorithm as following [7]:

1) Blue algorithm increases the packet loss when queue overflow, and decreases the packet loss when queue is idle. But queue overflow and idle are two extreme, so the queue length increases or decreases rapidly and the amplitude of fluctuation is very large. It will result in the decrease of throughput when the queue length fluctuates heavily. Moreover, the oscillation of queue length results in greatly end-to-end delay jitter so that it is difficult to meet the requirements of QOS.

2) It lacks adaptability in practical application and cannot make relevant adjustment according to the virtual

condition of the network, which is one of the reasons for causing great fluctuation of throughput and queue length.

2.3 AQM IMPLEMENTATION SCHEME

The common AQM Implementation Scheme is to append complex AQM algorithm to the Router, shown as Figure 2. In the Figure, Sourcer means sending end, Receiver means receiving end, Qs means queue scheduling, Queue means the queue. When new data packet arrives, the AQM algorithm will be run in the Router; and then it will decide whether to drop the data packet or queue it up according to the running result of the algorithm; at last, it will be trans ponded by look up routing list according to the routing information carried at the top of rejected packets and at the same time the router will save the intermediate variable of the algorithm for the use of operation next moment [8].

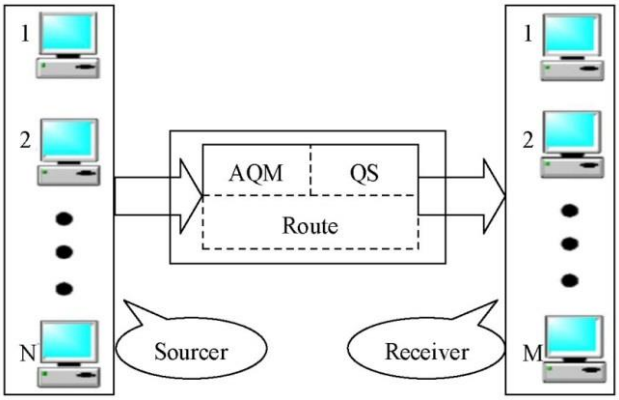

FIGURE 2 The implementation scheme of AQM algorithm

3 The design AQM algorithm and overall structure of performance detection platform

3.1 THE IMPLEMENTATION OF AQM ALGORITHM DETECTION PLATFORM

The Linux-based NS2 Platform can simulates the performance of different scale network algorithms, and the result is more close to practical effects, so this thesis designs a detection and performance evaluation platform for AQM a

Algorithm. The Figure 3 shows the AQM Mechanism Design and Detection Platform Framework based on NS2. As seen in the Figure 3, the reality of the platform can be mainly divided into several parts as follows:

1) AQM algorithm Engine Module: the AQM mechanism deployed at bottleneck router, which is executed frequently, and the execution efficiency will influence the control performance of the whole system's network congestion. The algorithm realized the functions as follows: it samples the link congestion condition periodically (based on timer module) and collects latent congestion variable in time via the congestion detection module of AQM network, and then updates packet loss dynamically via specific queue management engine; when every data packet reaches the router, if queue buffer memory overflows, then physical packet loss will be carried out; if buffer memory is not full, then packet loss event in probability of AQM algorithm will be triggered [9].

2) Network Topology Module: the module is mainly designed to achieve the layout of network topology scene. Topology includes single-bottleneck mode and multi-bottleneck mode. Single-bottleneck adopts "dumbbell-shaped" structure. Although each packet needs to go through multiple links from the source end to the terminal, as for each linking flow, among many links only one is the bottleneck. Hence, it can simplify the network as single-bottleneck link; multi-bottleneck topology adopts "parking lot" structure. When the algorithm is detected through single-bottleneck topology, the network topology will be complicated and multi-link mode will be introduced, which will make the algorithm results are more close to real network performance. Network topology module is a simulated network macroscopic module, which is defined during the simulation initialization process. Specific network topology will be generated to define network bandwidth for each link and define the link transmission delay time [10].

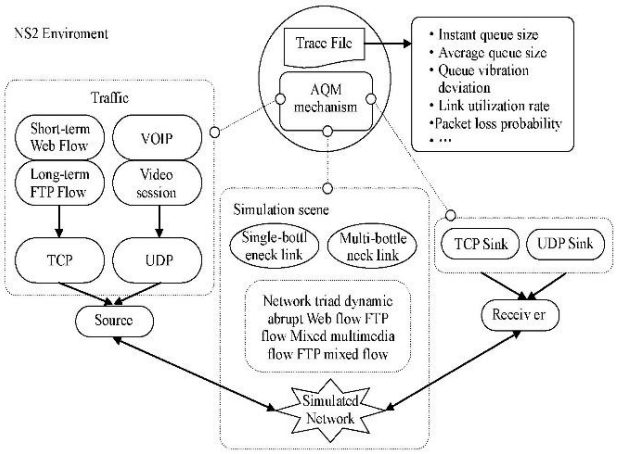

FIGURE 3 AQM algorithm detection platform based on NS2

3) Module for Abrupt Network: the module mainly manages the flow types and abrupt time. The flow types of internet are very abundant, but they are mainly consisted of long-term TCP flow (FTP), short-term TCP flow (HTTP) and response UDP flow (loading multimedia service). Network flow possesses inherent abruptness, which leads to that the network congestion is unforeseeable and difficult to control. Therefore, in order to verify the algorithmic comprehensive performance of AGM algorithm in facing complex network flows, the detection platform concentrates on detection the single abrupt case and mixed abrupt case of above flows so as to ensure that the simulation results can comprehensively reflect the performance of AQM algorithm and verify the disadvantages of this algorithm [11].

4) Module for Performance Analysis: the module mainly to achieve the function of intelligently analysing simulation results and extracting data relevant to AQM algorithm performance indicators according to simulation

results files (Trace File). Such data include instant queue size curve varying with time, average queue size, queue vibration variance, link utilization rate and packet loss probability. By analysing and comparing simulation results, it can reach relevant conclusions [12].

The following parts of the present research will develops by focusing on the specific designing of these modules.

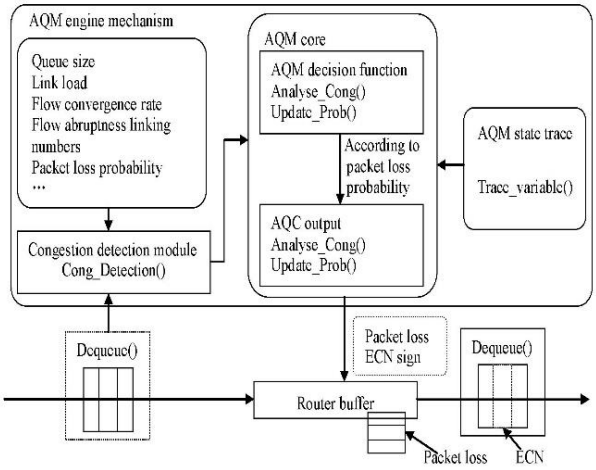

FIGURE 4 The Design of AQM Algorithm Engine Modules

3.2 THE DESIGN OF AQM ALGORITHM ENGINE MODULES

The unbalance of network flow distribution and the abruptness of data lead to the occurrence of network congestion. The AQM router placed in the scene, where congestion occurs, perceives the real-time network congestion through network flow changes and then utilizes AQM decision function to dynamically control packet loss probability. The router also makes the sending end learn the features of network congestions in advance through packet loss or ECN means and further achieve the reasonable utilization of network resource. AQM mechanism is the core module of congestion control. It needs to implement packet control for each arriving packets. Hence, it has high requirements for algorithm efficiency. As a result, here we adopt C++ language of NS2 to meet the requirement. Corresponding mapping is indicated as Queue/XXX. XXX is represented as the definition means of AQM. According to the above analysis, it can be known that a complete AQM mechanism (as shown in Figure 4) consists of five components: congestion detection module, AQM decision function module, AQM output control module, AQM state trace module and AQM engine message event management module.

1) Congestion detection module: the module mainly detects the congestion situation of current links. Different AQM mechanisms acquire different quantity of state, which mainly includes the following detection scales: instant queue size/average queue size (e.g. RED algorithm and PI algorithm), link load (e.g. Blue algorithm and Yellow algorithm) flow convergence rate (e.g. REM algorithm and VRC algorithm), active flow abrupt linking numbers (e.g. FRED algorithm and SBF algorithm) and packet loss probability (e.g. LRED algorithm). Many algorithms adopt mixed scales to comprehensive detect the network congestion situation. Among the above scales, apart from instant queue size and link load which can be directly acquired from the links, all other scale need to make periodical sampling congestion situation to conduct a statistics about the comprehensive information of network congestion during a certain period so as to filter the transient interference of sampling information.

2) AQM Decision Module: the module is the core of AQM engine. It has two major functions: one is the function of analyzing and acquiring potential congestion cases according to network congestion parameters; another is the function of designing specific queue control function according to analysis results and adjusting the packet loss value according to network congestion state.

3) AQM Output Control Module: the functional module mainly dynamically makes congestion marking and control on the packets it goes through according to the control output (packet loss value) of AQM decision function. On the premise of the support of AQM router and TCP, when the buffer of AQM router overflows, mandatory physical packet loss will be achieved; otherwise, according to packet loss probability value, the probability will set the CEN of IP head for the passing packets as 1 so as to realize the logical packet loss. If TCP does not support ECN mechanism, the transmission and delivery of congestion feedback information will be achieved by conducting physical packet loss in probability.

4) AQM State Trace Module: the module mainly functions to make file records for certain variables during simulation process and provide original data for AQM algorithm performance analysis module. The variable values that AQM algorithm needs to trace are defined in accordance with the demands of algorithm design. Generally speaking, variable values needed to record include instant queue size, average queue size, packet loss value, link load and other customized variables.

5) AQM Engine Message Event Management Module: the module mainly consists of several timers and is used to manage the triggering of periodical events. For instance, the active flow estimation and packet loss estimation of congestion control module require periodical sampling. Queue control function periodically updates the packet loss probability function value.

3.3 VQUDP PACKET LOSS STRATEGY MODULE

Firstly, based on the results of UDP active flow perceives by APF, we will calculate and weight the convergence rate (r_{id}) of every UDP in unit time. Therefore, the total convergence rate (R_{ater}) of UDP is:

$$R_{ater} = \sum_{jd \in L} r_{id} .$$

Among it, L is UDP flow collection (the BFV maintained by UDP flow). Calculate the current UDP rate perceive factor (λ) through Equation (3) and C is the actual bandwidth of the current bottleneck link (with its unit as Mpbs).

$$\lambda = \frac{R_{ater}}{C}. \tag{3}$$

UDP is mostly used in the transmission of network multi-media audio/video and certain network packet loss won't affect people's understanding of the information. At the same time, UDP is an open-loop control flow, which means that it cannot respond to the packet loss/ECN feedback signal caused by network congestion. Therefore, the UDP flow data packet is usually abandoned in the forced manner.

VQudp packet loss strategy uses the similar packet loss mechanism to Blue algorithm, which can be seen in Equation (4). In this Equation, *kudp* is the probability packet loss gain value; $qlen_{upd}$ is the instantaneous virtual queue length of UDP and VQ_{upd} is the logic size of the current UDP virtual queue buffer.

$$p = \max\left\{\frac{qlen_{upd}}{VQ_{upd}} \cdot k_{udp}, 1\right\}. \tag{4}$$

In order to make VQudp packet loss strategy can adjust its packet loss probability in face of different UDP loads, we have divided the network congestion caused by UDP into the following scenarios via UDP rate perceive factor (λ): when $\lambda \le 0.3$, UDP will lead to slight network congestion and when $\lambda > 0.7$, UDP will greatly deteriorate the network performance and cause network congestion. The above scenarios will adjust the packet loss gain according to the load value (λ) via Equation (5). When $\lambda > 0.3$, VQudp strategy uses the original packet loss gain (k_{upd}) and when $\lambda > 0.3$ We will adjust the packet loss gain (k_{udp}) dynamically according to the load (λ).

$$k_{upd}\begin{cases} k_{upd}\,\lambda \le 0.3 \\ \dfrac{k_{upd}}{1-\lambda}, \lambda > 0.3 \end{cases}. \tag{5}$$

3.4 VQTCP PACKET LOSS STRATEGY MODULE

The VQtcp packet loss strategy in this paper is based on the recently-raised Blue algorithm. In this paper, we have made some alterations to Blue algorithm and come up with a VBlue based on virtual queue to adapt to the VQtcp. Since TCP is the response flow of congestion control, it can adjust and control its sending rate according to the network congestion feedback signal. Therefore, when we design VQtcp packet loss strategy, we will use packet loss/ECN strategy instead of the forced physical packet loss, which is to say, when the virtual queue logic of VQtcp overflows, forced packet loss will be used; otherwise, ECN strategy will be used. VBlue algorithm has made the following changes to the original Blue algorithm:

1) On the basis of the original Blue algorithm, add the periodical update strategy of the size of virtual queue, namely after every period (ΔT), make BuffsizeVBlue=VQtcp via AFP algorithm;

2) Input the congestion control of the original Blue algorithm to make the instantaneous queue length (*qlen*) into (*qletcp*). Configure the target queue length of VBlue as $Q_{ref} = VQ_{tcp}/2$. VBlue algorithm takes the deviation (Δe) of the instantaneous queue length (*qletcp*) and the target queue length $Q_{ref} = VQ_{tcp}/2$ as the algorithm input. Δe represents the deviation degree between the instantaneous queue length and the target value.

3) In order to make the algorithm more flexible, VBlue algorithm first normalizes Δe to the range of [1-11] through the normalizing factor (*Gq*) and then normalize Δe to obtain the pack loss probability via Blue congestion control algorithm. The normalizing factor (*Gq*) here can be obtained from Equation (6).

$$G_q = \frac{2}{VQ_{tcp}}. \tag{6}$$

4 VFQ algorithm simulation analysis based on NS2 platform

This paper will design the following 5 congestion simulation scenarios (connection N mutations, dynamic link bandwidth C changes, dynamic round-trip time RTT, dynamic changes of non-response load λ & dynamic changes of Web loads P). To investigate the performance of these algorithms in the dynamic network status triad (N, RTT, C) and use the average queue length (*Avgq*), the queue oscillation variance (STD), link utilization (Utiliz) and packet loss rate (*LossR*) as the performance indexes.

4.1 THE ALGORITHM CHARACTERISTICS OF DIFFERENT CONNECTIONS

The continuous traffic change is the outstanding characteristics of computer network; therefore, the experiment is aimed to study the algorithm characteristics of VFQ algorithm under the ever-changing network congestion. Here, we only use TCP as the traffic of the experiments. In order to represent the dynamic changes of full-load network congestion, the active flow connections N will adopt the form of segmentation with time changes {300,600,200,700,300,500,200}, maintains every N for 50s and last the simulation for 350s.

The experimental results demonstrate that in most cases, ARED algorithm can control the instantaneous queue near the target value Q_{ref} =150pkt; however, when suffered from heavy load (N=700)0 queue oscillation is too serious to converge; the queue management performance of PI algorithm is the worst among 4 algorithms and it is impossible for it to converge qlen to the target value Q_{ref} =150pkt within 50s; the convergence

rate of Blue algorithm is relatively slow and it wobbles seriously with qlen; VFQ algorithm we come up with can capture the real-time network sudden changes through AFP active flow perceive algorithm. The actual value of VFQ active flow connections and the perceive value curve: AFP can converge the transient-state perceive value to the actual value 5s after every active flow change. Compared with other algorithms, VFQ algorithm has faster queue convergence rate and better queue control effects.

4.2 THE ALGORITHM CHARACTERISTICS OF DIFFERENT ROUND-TRIP TIME RTT

The scholars and engineers have verified the time-lag RTT parameter has a far-reaching influence on AQM algorithm from a great number of experiments. Our existing TCP/AQM system is in fact a time-lag RTT closed-loop control system with ECN/packet loss congestion. The reflected time lag is the packet RTT. Therefore, in order to verify that the robustness of VFQ algorithm in different RTT scenarios, this group of experiments have made repeated simulation to the network congestion scenarios when RTT = [60×i]ms (i =1, 2, ..., 7), N = 200 with other configuration as the default values.

ARED algorithm suffers most influence from RTT. When RTT≥240ms, its *qlen* is not stable, causing serious wobble; RTT plays little impact on *Avgq* and STD of PI and VFQ algorithms; the queue management of Blue algorithm is placed in the middle, but when RTT≥300m, its STD is also big. When RTT > 180ms, the link utilization Utiliz of ARED algorithm falls sharply, while the Utiliz performances of other various algorithm are almost the same. From the relationship between the packet loss probability *LossR* of various AQM algorithms and RTT, it is clear that the *LossR* of ARED and PI algorithms are at least one quantative grade higher than that of Blue and VFQ algorithms. To summarize, VFQ algorithm has better stability and robustness in the scenarios with RTT changes.

4.3 THE ALGORITHM CHRACTERISTICS OF DIFFERENT LINK BANDWIDTH C

The bandwidth *C* determines the available resources of the bottleneck link. In general, *C* in the wired network is usually fixed while it is instable in the wireless network or satellite link due to the environmental influence. Therefore, this part will verify the performances of various AQM algorithms in different bandwidth *C* on NS2 simulation platform in the network topology of Figure 1. Repeated simulation is made to the congestion network scenario when *C*={5,10,15,25,35,45,55} Mbps, RTT=80ms, N=300.

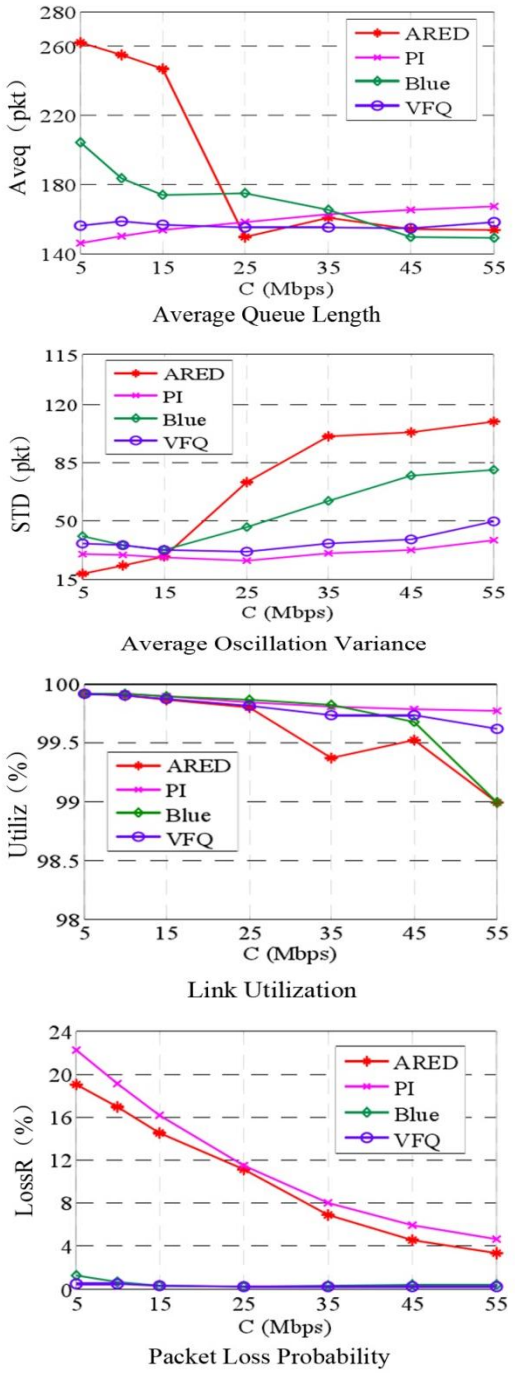

FIGURE 5 The comparisons of different algorithm performances under the changes of dynamic bandwidth C

Figure 5 describes the changes of various characteristic indexes of AQM in different network bandwidth *C*. On the whole, with the continuous changes of bandwidth *C*, the queue oscillation STD will increase gradually and the network utilization Utiliz decreases. So do *Avgq* and Loss R.

When C≤25Mbps, the *Avgq* of ARED algorithm is very big; however, with the gradual increase of the bandwidth *C*, *Avgq* gets closed to the target value *Qref* (150pkt). It is clear in Figure 5b that when *C* > 35Mbps, STD increases, meaning that ARED algorithm oscillates dramatically.

Both VFQ and PI algorithms have made impressing results in the queue management and bandwidth resource utilization; however, in Figure 5d, we can see that the *LossR* of PI algorithm and ARED algorithm are one quantative grade than that of VFQ algorithm. In one word, VFQ algorithm has better performance in the scenarios with dynamic changes in bandwidth C.

4.4 THE ALGORITHM CHARATERISTICS OF THE LOADS λ IN DIFFERENT NON-RESPONSE FLOW

If we want to verify the performance of the above-mentioned AQM algorithms in the computer network scenarios with mixed flow of TCP/UDP through NS2 simulation platform, we need to examine the UDP flow load λ to every AQM algorithm characteristic. Repeated simulation will be made to various AQM algorithms in TCP connections N = 400, λ={0.05,0.15,0.25,0.35,0.45,0.55,0.6} and RTT = 80ms.This is the relationship between UDP proportion factor λ and the performance indexes of various AQM algorithms. ARED algorithm has similar *Avgq* curve to Blue algorithm; however, STD curve of ARED algorithm wobbles little in contract with that of Blue algorithm. When λ=0.05, *Avgq* of PI algorithm is big and the queue becomes instable (*Avgq* deviates from Qref; STD increases) gradually with the increase of (λ). VFQ algorithm can stabilize *qlen* near the target area, although STD increases with the increase of UDP proportion factor λ. Λ has placed the biggest influence on Utiliz of PI algorithm, while the Utiliz of other algorithms can almost reach full-load 100% status. From the perspective of the packet loss probability, the *LossR* of ARED and PI algorithms have been over 13%, while the *LossR* of Blue and VFQ algorithms increases greatly with the increase of the load λ. The *LossR* increases 8~9 times when λ=0.05 and λ=0.6. To sum up, non-response UDP load λ has a big influence on the performance indexes of various AQM algorithms; however, compared with other algorithms, although the *LossR* of VFQ algorithm also suffers big impact, it can still maintain *qlen* in the target area Qref.

4.5 THE ALGORITHM CHARACTERISTICS OF DIFFERENT WEB LOADS P

This part verifies the above AQM algorithms performance in the long-time TCP/Web traffic mixed-flow network congestion scenario through NS2 simulation platform. It adopts PackMime model by Bell Laboratory as the generative model of Web traffic and uses HTTP/1.1 version protocol as the data communication protocol carrier of short-time Web flow, among which, ρ is the requests of Web traffic per second (connections). The simulation will be made to the AQM algorithms in the network scenarios when ρ={10,20,30,40,50,60,70} (connection/seconds), RTT=80ms and the long-time TCP connections N=300.

The variation trends of different performance indexes of various AQM algorithms in different Web traffic loads. Although the *Avgq* of ARED and Blue algorithms keep stable with the changes of load ρ, it is clear from the STD curve that the queue wobbles (around 80pkt), meaning that ARED and Blue algorithms can't get satisfactory queue management performance in dealing with TCP/Web mixed-flow scenario. Although PI algorithm has a relatively stable STD curve, its *Avgq* is around 250pkt most of the time and it cannot converge *qlen* to near Q *ref*=150pkt. The VFQ algorithm proposed by this paper adopts AFP active flow perceive algorithm, making it better adapt to the changes of Web flow load ρ and its overall queue performances suffer little influence. Apart from PI algorithm, the Utiliz of other algorithms fall with the increase of Web load ρ. With the increase of Web load ρ, the *LossR* of various algorithms increase while the *LossR* of PI algorithm demonstrates an instable status. To sum up, the addition of Web traffic makes the network congestion scenario more complicated and causes bigger influence on the traditional AQM algorithms, which only take TCP into consideration. n the meanwhile, it can be seen from this group of simulations that, in the network congestion scenario of TCP/Web mixed flow, although various performances of VFQ algorithm fall, it receives the least influence compared with other AQM algorithm.

5 Conclusions

The present research conducts a detailed research on AQM algorithm. Based on Linux system, the present research establishes an AQM algorithm detection platform of NS2, which can simulate the performance of network algorithms with different scales. Moreover, the present research also designs an AQM algorithm engine module, on the basis of which, it raises Multi-Virtual Queue based on Flow Estimation (VFQ). Finally, this paper has also conducted plenty of simulation to VFQ algorithm, made some simulation comparisons with some classical AQM algorithms such as ARED algorithm, PI algorithm and Blue algorithm and analysed the performance of VFQ algorithm in order to verify the feasibility and validity of the new algorithm. It is demonstrated in the simulation results that in face of the dynamic load changes, VFQ algorithm has a rapid convergence rate and high-efficient congestion queue processing ability in contrast with other algorithms, suggesting that VFQ algorithm can better deal with the network congestion.

Acknowledgments

This paper comes from the research fund of Heze University (No. XY12KJ03).This work was supported by the science and technology projects of the Shandong province universities (No. J12LN55, No. J13LN53).

References

[1] Li F, Sun J, Zukerman M, Liu Z, Xu Q, Chan S, Chen G, Ko K-T 2014 A comparative simulation study of TCP/AQM systems for evaluating the potential of neuron-based AQM schemes *Journal of Network and Computer Applications* **41** 274-99

[2] Hwang L-C 2013 M-GREEN: An active queue management mechanism for multi-QoS classes *Computer Standards & Interfaces* **36(**1) 122-31

[3] Divakaran D M 2012 A spike-detecting AQM to deal with elephants *Computer Networks***56**(13) 3087-98

[4] Guesmi H, Djemal R 2013 Design of priority-based active queue management for a high-performance IP switch *Computers & Electrical Engineering* **39**(2) 246-60

[5] Zhang W, Tan L, Yuan C, Chen G, Ge F 2013 Internet primal–dual congestion control: Stability and applications *Control Engineering Practice* **21**(1) 87-95

[6] Zhan Z-q, Zhu J, Xu D 2012 Stability analysis in an AVQ model of Internet congestion control algorithm *The Journal of China Universities of Posts and Telecommunications* **19**(4) 22-8

[7] Avrachenkov K, Ayesta U, Doncel J, Jacko P 2013 Congestion control of TCP flows in Internet routers by means of index policy *Computer Networks* **57**(7) 3463-78

[8] Zhou H, Hu C, He L 2013 Improving the efficiency and fairness of eXplicit Control Protocol in multi-bottleneck networks *Computer Communications* **36**(10–11) 1193-208

[9] Zhang C, Cai Z, Chen W, Luo X, Yin J 2012 Flow level detection and filtering of low-rate DDoS *Computer Networks* 56(15) 3417-31

[10] Yilmaz M, Ansari N 2014 Achieving destination differentiation in ingress aggregated fairness for resilient packet rings by weighted destination based fair dropping *Computer Networks* **62** 43-54

[11] Giaccone P, Leonardi E, Neri F 2013 On the interaction between TCP-like sources and throughput-efficient scheduling policies *Performance Evaluation* **70**(4) 251-70

[12] Ju Y, Wang A, Liu X 2012 Evaluating emergency response capacity by fuzzy AHP and 2-tuple fuzzy linguistic approach *Expert Systems with Applications* **39**(8) 6972-81

Author

Wenqing Huang, born in 1981, Heze, Shandong, China

Current position, grades: an assistant of the Department of Computer and Information Engineering, Heze University.
University studies: Master of Engineering in Computer Science from the Liaoning University of Science and Technology in 2011.
Scientific interest: computer network, network security.