International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

An Effectiveness Service to Manage Traffic in High Speed Networks

S. Shafiulla[1], K. Ramadevi[2]

[1]PG Student, Annamacharya Institute of Technology and Sciences, Kadapa, Tirupati, India

[2]Assistant Professor, Annamacharya Institute of Technology and Sciences, Kadapa, Tirupati, India

Abstract: ***In view of the fast-growing Internet traffic, this paper propose a distributed traffic management framework, inwhich routers are deployed with intelligent data rate controllers to tackle the traffic mass. Unlike other explicit traffic control protocols that have to estimate network parameters (e.g., link latency, bottleneck bandwidth, packet loss rate, or the number of flows) in order to compute the allowed source sending rate, our fuzzy-logic-based controller can measure the router queue size directly; hence it avoids various potential performance problems arising from parameter estimations while reducing much consumption of computation and memory resources in routers. As network parameter, the queue size can be accurately monitored and used to proactively decide if action should be taken to regulate the source sending rate, thus increasing theresilience of the network to traffic congestion. The communication QoS (Quality of Service) is assured b y the good performances of our scheme such as max-min fairness, low queuing delay and good robustness to network dynamics. Simulation results and comparisons have verified the effectiveness and showed that our new traffic management scheme can achieve better performances than the existing protocols that rely on the estimation of network parameters.***

Keywords: Congestion control, Fuzzy logic control, Quality of service, Max-min fairness, Traffic management.

1. Introduction

Network traffic management can prevent a network from severe congestion and degradation in throughput delay performance. Traffic congestion control is one of the effective approaches to manage the network traffic. Historically, TCP (Transmission Control Protocol) Reno is a widely deployed congestion control protocol that tackles the Internet traffic. It has the important feature that the network is treated as a black box and the source adjusts its window size based on packet loss signal .However, as an implicit control protocol, TCP encounters various performance problems (e.g., utilization, fairness and stability) when the Internet BDP (Bandwidth-Delay Product) continues to increase These have been widely investigated with various proposed solutions such as the AQM(Active Queue Management).

FLC (Fuzzy Logic Control) has been considered for IC (Intelligence Control). It is a methodology used to design robust systems that can contend with the common adverse synthesizing factors such as system nonlinearity, parameter uncertainty, measurement and modeling imprecision. In addition, fuzzy logic theory provides a convenient controller design approach based on expert knowledge which is close to human decision making, and readily helps engineers to model a complicated non-linear system. FLC has found its applications to network congestion control. In early stage, it was used to do rate control in ATM network to guarantee the QoS.

Specifically, the objectives of this paper are: 1) to design a new rate-based explicit congestion controller based on FLC to avoid estimating link parameters such as link bandwidth the number of flows, packet loss and network latency, while remaining stable and robust to network dynamics ,2) to provide max- min fairness to achieve an effective bandwidth allocation and utilization 3) to generate relatively smooth source throughput maintain a reasonable network delay and achieve stable jitter performance by controlling the queue size, 4) to demonstrate our controller has a better QoS performance through case study. The contributions of our work lie in: 1) using fuzzy logic theory to design an explicit rate-based traffic management scheme (called the IntelRate controller) for the high-speed IP networks; 2) the application of such a fuzzy logic controller using less performance parameters while providing better performances than the existing explicit traffic control protocols; 3) the design of a Fuzzy Smoother mechanism that can generate relatively smooth flow throughput; 4) the capability of our algorithm to provide max-min fairness even under large network dynamics that usually render many existing controllers unstable.

2. Traffic Management Principle and Modeling

We consider a backbone network interconnected by a number of geographically distributed routers, in which hosts are attached to the access routers which cooperate with the core routers to enable end-to-end communications. Congestion occurs when many flows traverse a router and cause its IQSize (Instantaneous Queue Size) to exceed the buffer capacity, thus making it a bottleneck in the Internet. Since any router may become bottle neck along an end -to-en d data path, we would like each router to be able to manage its traffic. If the former is smaller

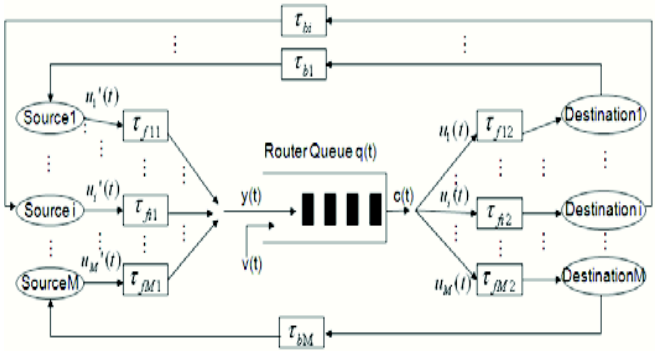

Figure 1: System model of an AQM router

than the latter, the *Req_rate*field in the packet header will be updated; otherwise it remains unchanged. After the packet arrives at the destination, the value of the *Req_rate*field re ects the allowed data rate from the most congested router along the path if the value is not more than the desired rate of the source.

3. The IntelRate Controller design

Figure 2 depicts the components of our fuzzy logic traffic controller for controlling traffic in the network system defined in Fig. 1. Called the IntelRate, it is a TISO (Two-Input Single- Output) controller. The TBO (Target Buffer Occupancy) 0 is the queue size level we aim to achieve upon congestion. The queue deviation e(t)=q0-q1(t)) is one of the two inputs of the controller.

FLC is a non-linear mapping of inputs into outputs, which consists of four steps, i.e., rule base building, fuzzification, inference and defuzzification. The concepts of fuzzy set and logic of FLC were introduced in 1965 by Zadeh, and it was basically extended from two-valued logic to the continuous interval by adding the intermediate values between absolute TRUE and FALSE. In the sequel, we formulate our new con-troller by following those four steps along with designing the fuzzy linguistic descriptions and the membership functions. The parameter design issues and the traffic control procedure are also discussed at the end of the section.

A)Linguistic description and rule base:

We define the crisp inputs e(t),g(e(t)) and output u(i) with the linguistic variables e(t),g(e(t)) and u(t).

Table 1: Rule Table for Intel Rate Controller

Allowed Throughput *u(t)*		*e(t)*								
		NV	NL	NM	NS	ZR	PS	PM	PL	PV
g(e(t))	NV	ZR	ZR	ZR	ZR	ZR	ES	VS	SM	MD
	NL	ZR	ZR	ZR	ZR	ES	VS	SM	MD	BG
	NM	ZR	ZR	ZR	ES	VS	SM	MD	BG	VB
	NS	ZR	ZR	ES	VS	SM	MD	BG	VB	EB
	ZR	ZR	ES	VS	SM	MD	BG	VB	EB	MX
	PS	ES	VS	SM	MD	BG	VB	EB	MX	MX
	PM	VS	SM	MD	BG	VB	EB	MX	MX	MX
	PL	SM	MD	BG	VB	EB	MX	MX	MX	MX
	PV	MD	BG	VB	EB	MX	MX	MX	MX	MX

B) MembershipFunction, Fuzzification and Reference

Our IntelRate controller employs the isosceles triangular and trapezoid -like functions as its MFs (Membership Functions). Figure 3 describes the MFs used to determine the certainty of a crisp input or output. To determine how much a rule is certain to a cur- rent situation, our controller applies the Zadeh AND logic [46] to perform the inference mechanism, e.g. for crisp inputs p1 and p2 , the final certainty (also referred to as the firing level).

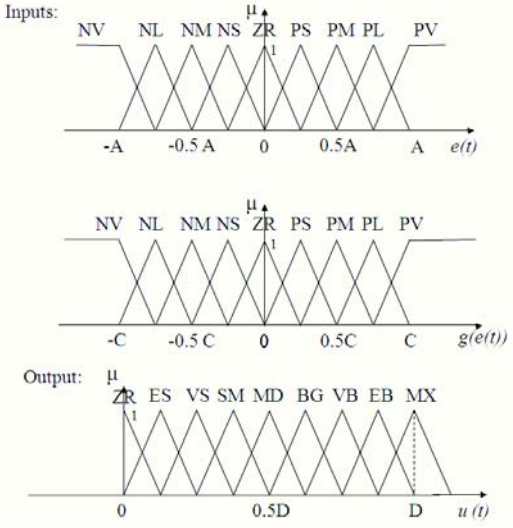

Figure 2: Architecture of general FS

While Fig. 4 is used to illustrate the design of a general FS, the designated values actually come from an example of N=9LVs with the absolute values of both the upper and lower limits of g(e(t)) set to mq0 .Since e(t) is bounded by the physical size of a queue, we have the boundaries according to the limits q0-B=e(t)=q0

C) Defuzzification

For the defuzzification algorithm, the IntelRate controller applies the COG (Center of Gravity) method.

D) Design parameters

From the perspective of the queueing delay, the TBO value should be as small as possible.

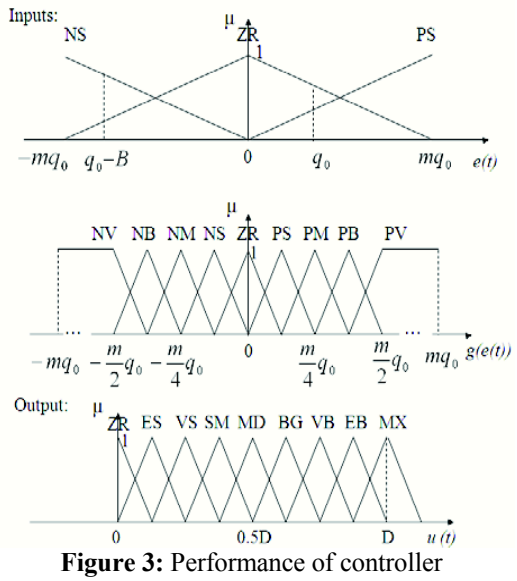

Figure 3: Performance of controller

From our design above, one can see there are different parameters which ultimately will affect the performance of our traffic controller.

E) Control procedure

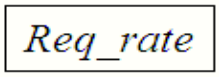

Figure 5 shows the new field in the packet congestion header that we need to support our controller algorithm for the operation principle mentioned in Section II. As discussed, we need to include in the congestion header a new field called req_rate to carry the desired sending rate from the source and which will be continuously updated by the allowed sending rate when the packet passes each router Below is a summary of the traffic-handling procedure of the IntelRate controller in a router.

1) Upon the arrival of a packet, the router extracts Req_ratefrom the congestion header of the packet.
2) SampleIQSize q(t) and update e(t) and g(e(t)).
3) Compute the output u(t) and compare it with Req_rate
   a) If u(t)<Req_rate , it means that the link does not have enough bandwidth to accommodate the requested amount of sending rate. The Req_rate field in the congestion header is then updated by u(t).
   b) Otherwise the Req_rate remain unchanged.
4) If an operation cycle *d* is over,the crisp output u(t) and the output edge value of D.

4. Performance Evaluation

The capability of the IntelRate controller is demonstrated by performance evaluations through a series of experiments.

A) Simulated Network:

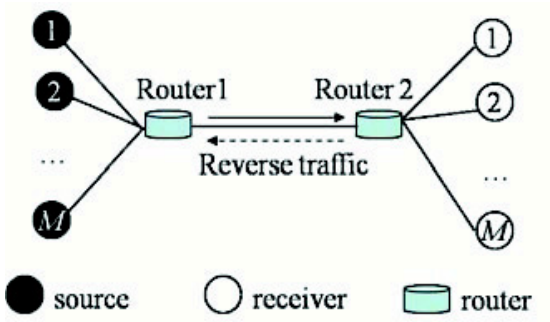

Figure 4: Simulation network.

The single bottleneck network in Fig.6 is used to investigate the controller behavior of the most congested router. We choose Router 1 as the only bottleneck in the network, whereas Router 2 is configured to have sufficiently high service rate and big buffer *B* so that congestion never happens there.In order to demonstrate and discuss the robustness of our IntelRate controller, our experiments would focus on the testing of the 100 long-lived ftpflows, unless otherwise stated.

The experiments were conducted using OPNET Modeler on an Intel Core TM2 Quad platform with 2.40GHz processor. Typical simulated time is set according to the specific experiment scenario. The simulation time depends on bottleneck bandwidth and the simulated time. A typical simulation run usually takes hours or days. For example, in order to observe the source throughput behavior before and after the network parameter change, we set a longer simulated time for such an experiment than a max-min fairness experiment. The number of packets generated in an experiment is related to th e TBO value, the bandwidth, the simulated time and the traffic intensity. The controller is evaluated by the following performance measures.

a) Source throughput (or source sending rate) is defined to be the average number of bits successfully sent out by a source per second, i.e. bits/second . Here, a bit is considered to be successfully sent out if it is part of a packet that has been successfully sent.
b) Queueing delay is the waiting time of a packet in the router queue before its service.

B) Robustness to large network changes:

The real world Internet traffic is always dynamic. Below we shall investigate the performance of our controllers when faced with drastic network changes such as the variations of the number of rows or the available bandwidth.

C) Sudden traffic changes

Sudden traffic change may occur when the number of long-lived or short-lived flows, or some uncontrolled (unresponsive) flows change at the ingress/egress. To test the controller capability to handle the dynamics of the real Internet traffic, we conduct an experiment in which the bottleneck bandwidth is 3Gbps. TBO is set to be 3000 packets, which results in a queuing delay of about 8.19ms when the queue size stabilizes at TBO

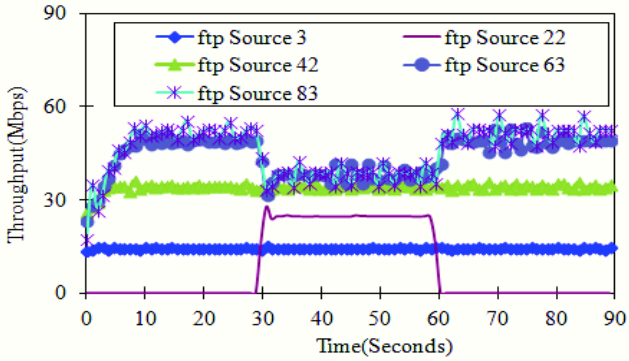

Figure 5: Source and IQSize dynamics under traffic change.
Figure 9 (a) shows the throughput dynamics of 5 representative ftp sources. The sources in groups 1 and 3 can maintain relatively constant throughput.

5. Comparison

Our preliminary results demonstrate that the IntelRate controller is superior to other explicit congestion controllers. We use the same IEEE 802.11 wireless LAN (Local Area Network) as done in QFCP and Blind to do the comparison. We use the Application and Profile module of the OPNET to generate traffic in the sources. The TBO of the IntelRate controller is set to 60 packets. The IntelRate controller

presents a much more stable queue size after it quickly reaches the TBO of 60 packets

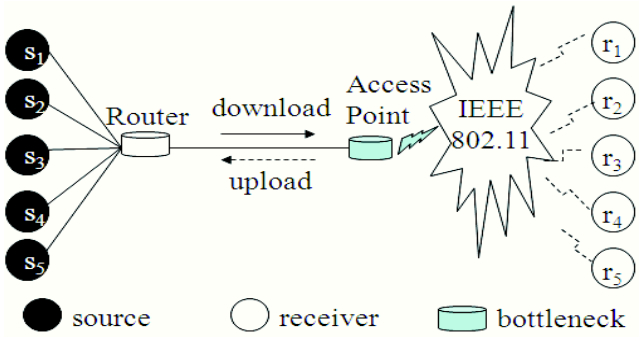

Figure 6: Wireless LAN

6. Conclusion

A novel traffic management scheme, called the Intel Rate controller, has been proposed to manage the Internet congestion in order to assure the quality of service for different service applications. A novel traffic management scheme, called the IntelRate controller, has been proposed to manage the Internet congestion in order to assure the quality of service for different service applications. Unlike the existing explicit traffic control protocols that potentially suffer from performance problems or high router resource consumption due to the estimation of the network parameters, the IntelRate controller can overcome those fundamental deficiencies. To verify the effectiveness and superiority of the Intel Rate controller, extensive experiments have been conducted in OPNET modeler

References

[1] M. Welzl, Network Congestion Control: Managing Internet Traffic. John Wiley & Sons Ltd., 2005.
[2] R. Jain, “Congestion control and traffic management in ATM networks: recent advances and a survey,” Computer Networks ISDN Syst., vol. 28, no. 13, pp. 1723–1738, Oct. 1996.
[3] V. Jacobson, “Congestion avoidance and control,” in Proc. 1988 SIGCOMM, pp. 314–329.
[4] V. Jacobson, “Modified TCP congestion avoidance algorithm,” Apr.1990.
[5] K.K. Ramakrishnan and S. Floyd, “Proposals to add explicit congestion notification (ECN) to IP,” RFC 2481, Jan. 1999.
[6] D.Katabi, M. Handley, and C. Rohrs, “Congestion control for high bandwidth-delay product networks,” in Proc. 2002 SIGCOMM, pp. 89–102.