一种针对组播的分布式自调节显式速率控制器*

谭连生+, 刘 芹, 余一娇

(华中师范大学 计算机科学系,湖北 武汉 430079)

A Distributed Self-Tuning Explicit Rate Controller for Multicast Flows

TAN Lian-Sheng+, Liu Qin, YU Yi-Jiao

(Department of Computer Science, Central China Normal University, Wuhan 430079, China)

+ Corresponding author: Phn: +86-27-67867651, Fax: +86-27-67863612, E-mail: l.tan@ccnu.edu.cn

Received 2003-06-03; Accepted 2004-01-06

Tan LS, Liu Q, Yu YJ. A distributed self-tuning explicit rate controller for multicast flows. *Journal of Software*, 2004,15(6):940~948.
http://www.jos.org.cn/1000-9825/15/940.htm

Abstract: The ever-increasing multicast data applications recently have aroused considerable interests in the design of congestion control scheme for multicast services. This kind of study is indeed important, especially to those multicast receivers with large propagation delays which mean the feedbacks arriving at the source node are somewhat outdated and harmful to control actions. A distributed self-tuning explicit rate algorithm is presented in this paper to overcome the vulnerability that suffers from the heterogeneous multicast receivers. It is suggested that congestion controllers be located at the source and the participating intermediate nodes to regulate the transmission rate. This network-assisted property is different from the traditional control scheme in that the router computes the appropriate transmission rate of itself and executes it rather than sends packets in best efforts. This active manner makes the control more responsive to the network status. The proposed self-tuning controller has essentially a proportional controller structure. The proportional gain is related to the extent that the router buffer occupancy deviates from the desired point. Simulation results show the efficiency of the proposed scheme in terms of fast response, high link utilization, and relatively stable buffer occupancy.

Key words: distributed algorithm; explicit rate; multicast congestion control; rate-based congestion control; self-tuning controller

* Supported by the National Natural Science Foundation of China under Grant No.60174043 (国家自然科学基金); the Key Project of Natural Science Foundation of Hubei Province of China under Grant No.2002AB025 (湖北省自然科学基金)

TAN Lian-Sheng was born in 1965. He is a full professor now and the Head of the Department of Computer Science, Central China Normal University. Dr. Tan obtained his Ph.D. degree from Loughborough University in UK in 1999. He was a postdoctoral research fellow doing researching in computer networks with School of Information Technology and Engineering, University of Ottawa, Canada in 2001. His research interests are in modeling, congestion control analysis, and performance evaluation of computer communication networks. **LIU Qin** is a MSc. candidate in Department of Computer Science, Central China Normal University. She is interested in computer network congestion control and traffic modeling. **YU Yi-Jiao** was born in 1978. He is currently a research fellow in Central China Normal University and obtained his Master degree of Computer Science from Central China Normal University. His research interests focus on computer network, artificial intelligence, and network management.

摘　要: 拥塞控制是组播服务需要解决的重要问题.当存在大量异质的、传播时延较高的组播接收者时,到达源端的反馈在一定程度上已经过时,因此常常严重影响控制效果.提出了一种分布式、自调节的显式速率比例控制方案,它在源端和中间节点实施控制算法调整各自的发送速率,并且比例控制系数会根据路由器缓冲区占有量偏离理想值的程度自动调节.这种网络参与控制的主动行为比传统路由器尽力服务的工作方式对网络状态的响应更加迅速,自调节的控制系数比固定的控制系数更能及时调节发送速率.仿真实验结果表明,算法具有响应快、链路利用率高和路由器缓冲区占有量比较稳定的特性.

关键词: 分布式算法;显式速率;组播拥塞控制;基于速率的拥塞控制;自调节控制器
中图法分类号: TP393　　**文献标识码**: A

1 Introduction

Multicast improves the efficiency of multipoint data distribution by building a distribution tree from a sender to a set of receivers[1]. However, the widely used multicast transport protocols which are layered on the top of IP multicast, could cause congestion or even congestion collapse if they do not provide adequate congestion control. Congestion control thus plays an important role in traffic management of multicast communications. There are many congestion schemes handling unicast transmissions efficiently[2~4]. Unfortunately, multicast congestion control is much more sophisticated than that of unicast due to the complexity of multicasting mechanism. Several multicast congestion approaches have been proposed recently. One class of them[5,6] adopts a simple hop-by-hop feedback mechanism, in which the feedback, i.e., backward control packets, from downstream nodes are initially gathered at branch points, and then are transmitted upwards by a single hop upon receipt of a forward control packet. The main merit of these methods lies in their simplicity of hop-by-hop mechanism, but at the same time they often lead to the so-called *consolidation noise problem*[7] due to incomplete feedback information. To overcome this drawback, Ref.[8] proposed a method called feedback synchronization that certain manipulations are performed at each branch point by accumulating feedbacks from all downstream branches. This scheme then introduces another problem of slow response due to the delay of feedback from “long” path. Such delayed congestion feedback can cause excessive queue build-up and packet loss at the bottleneck link. The authors of Refs.[9,10] suggested that only the suitable set of representatives instead of all receivers send their feedbacks to their sender. The authors of Ref.[11] proposed a fuzzy-logic-based consolidation algorithm to estimate the unknown congestion information caused by long propagation delay. More recently, Ref.[12] proposed an optimal second-order rate control algorithm to deal with control packet round-trip time (RTT) variations in multicast communications. This method has studied the system dynamics by using the binary congestion feedback in the scenarios of both persistent and on-off elastic traffic services, which defines that the data transfer rate is adjusted at the source depending on the available bandwidth at the bottleneck.

The major difficulty in the design of multicast congestion control protocols arises from the long and heterogeneous RTTs involved in the closed-loop control. The responsiveness of a congestion control scheme is crucial to how a protocol affects the network stability[13]. Concerning this regard, with a comparison to the binary feedback congestion control, the explicit rate scheme is more responsive to network congestion and can better serve wide area networks (WAN) environments where the bandwidth delay product is usually large. Explicit rate approach (see, for example Ref.[2]) has been proposed for unicast transmission systems. However, few attentions have been paid to the explicit rate formulations in multicast cases.

This paper develops a distributed self-tuning explicit rate algorithm to overcome the vulnerability that suffers from the heterogeneous multicast receivers. In our scheme, congestion controllers are located at the source and the participating intermediate nodes, i.e., the non-leaf nodes in a multicast tree, to regulate the transmission rate. This network-assisted property is different from the traditional control scheme in that the router computes the appropriate transmission rate of it and executes it rather than sends packets in best efforts. This active manner makes the control more responsive to the network status. In addition, the proposed self-tuning controller is essentially a proportional controller. The proportional gain is related to the extent that the router buffer occupancy deviates from the desired point. Therefore, the feedback consolidation problem is solved naturally within this algorithm. Each branch point only receives feedbacks from the direct downstream nodes instead of all downstream nodes; it thus greatly decreases the number of feedbacks that need to be processed at one node. As a result, our scheme can avoid *feedback explosion*[14] to a great extent. Simulation results show the efficiency of the proposed scheme in terms of fast response, high link utilization, and relatively stable buffer occupancy.

The paper is organized as follows. In Section 2, we present the overall architecture of the proposed multicast control scheme. We then evaluate the performance of the algorithm via various simulations in Section 3. It is finally concluded in Section 4.

2 Description of the Scalable Self-Tuning Rate Controller

2.1 System configuration

A rate-based congestion control algorithm is a feedback-based flow control mechanism for elastic traffic. Traditionally, the packet admission rate of the source is adjusted according to network status information carried in the feedback. Each intermediate node just sends packets received in best efforts without regulating its transmission rate. That is to say the link utilization is full if the link buffer is not empty. There is no doubt that this mechanism can make a good use of the network resources, but the passive manner will aggravate the downstream bottleneck congestion level in case there is a long distance between these two nodes. If rate regulation is only carried out at the source node, it has elapsed a long time from the congestion spot to the source. During the non-regulating period, the upstream node still emits packets at its maximum capacity, which will deteriorate the downstream bottleneck. In fact, we can alleviate the congestion level of the downstream node by shortening the duration of rate updating. Suppose that two adjacent nodes constitute a virtual “source-destination” pair, in which the upstream node acts as the source and the downstream one acts as the destination. The upstream node gets feedbacks from the direct downstream nodes and adjusts its transmission rate. In this way, we can deploy the same rate regulation mechanism in the real source node and the intermediate nodes.

As shown in Fig.1, the considered multicast elastic service in the network-assisted environments is described as follows.

(i) The network is a connection-oriented one and time is slotted by the sampling period with the duration $[n,n+1]$ equal to T. The associated data are transferred by a fixed size packet, called a data packet.

(ii) The source of a multicast session issues and transmits forward control packet (FCP) every sampling period in order to communicate flow-control related information with routers in the multicast tree.

(iii) The branch point of the multicast tree replicates each data packet and FCP from its upstream node to all its downstream branches. The downstream node returns its congestion information via backward control packet (BCP) to the parent through the backward direction of the coming path once it receives a FCP. Assume that congestion never happens at the router connected with the source, hence these two can be consolidated into one node, which is true in most cases in real networks. Under this assumption, the multicast source can also be treated in the same way as the branch point shown in Fig.1.

(iv) The buffer occupancy of the ith node is denoted by $x_i(n)$ at time slot n and the desired buffer level is denoted by $\bar{x}_i$. The router has sufficiently large storing capacity B, $0 << B < \infty$.

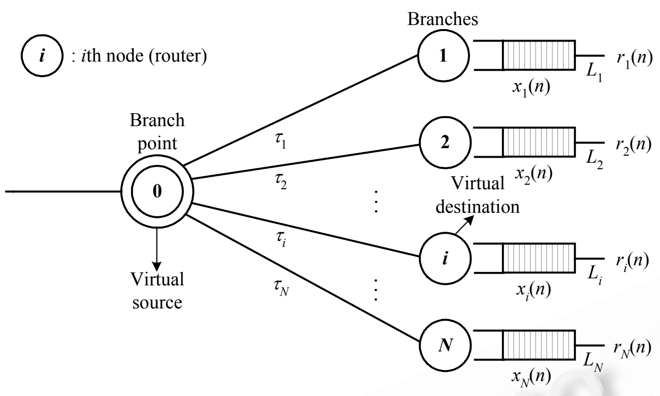


Fig.1 A multicast configuration at a branch point

(v) The ith link and its corresponding capacity is denoted by L_i without making confusion in the context. The time it takes for a packet to go from one end of the link to the other end (either in forward or in backward direction) is denoted by integer τ_i, which includes queuing delay, processing delay, and propagation delay. If the sum of these delays is not a multiple of T, it is sound to add a small value to the path delay to make τ_i integer.

(vi) Each router schedules the packets in a first-come-first-serviced manner. The component $r_i(n)$ represents the transmission rate of the ith node at time slot n. We use Fig.1 to describe the considered multicast model. With refer to Fig.1, the buffer occupancy of the ith node is determined by

$$x_i(n+1)=Sat_B\{x_i(n)+r_0(n-\tau_i)-r_i(n)\} \tag{1}$$

where $Sat_B\{x\}=\begin{cases}B & x>B\\ x & 0\le x\le B\\ 0 & x<0\end{cases}$ and $r_i(n)<L_i$.

If $Node0$ happens to be the multicast session source, another condition $MDR\le r_0(n)\le PDR$ must be satisfied, where MDR is the minimum data rate of the multicast session and PDR is the peak data rate.

2.2 The algorithm

The router's buffer occupancy is expected to stay at the neighborhood of the desired level. If $x(n)$ is too high, it often leads to buffer overflowing and packet loss. In addition, under this circumstance long packet queuing delay usually results in time out and retransmission, which in turn builds up the mounting of the buffer occupancy; consequently a vicious circle is formed. If $x(n)$ is too low, it increases the likelihood of link underutilization during the occasionally idle period. Thus the router buffer occupancy plays an important role in the congestion control that is chosen out to be the feedback carried in BCP. Generally, among all downstream nodes, the most congested one defined as the worst node deserves special attention. Based on this consideration, we propose the following proportional control scheme

$$r_i(n+1)=r_i(n)-C_p[x_{worst}(n-\tau_{worst})-\bar{x}_{worst}] \tag{2}$$

where C_p is the proportional gain, and $x_{worst}(n)$, $\bar{x}_{worst}$, τ_{worst} are the buffer occupancy, desired level and corresponding propagation delay of the worst node respectively. The component C_p can be carefully selected to ensure the stability of the system that guarantees the bounded buffer occupancy. However, the selection is generally difficult. Sometimes there even doesn't exist such a C_p at all. What's more, the fixed C_p can hardly capture the congestion level accurately once the session is established. If $x(n)$ leaves far away from the desired point $\bar{x}$, it is expected to bring the operating point quickly to the equilibrium value. Especially when $x(n)$ is much bigger than $\bar{x}$, a large C_p will prevent the node from dumping in congestion. When the system switches to the neighborhood of the desired point, it is expected that C_p come down to a small value to act conservatively. This observation motivates us to take a different method in choosing the feedback gain C_p from the conventional method like additive increase multiplicative decrease (AIMD) algorithm (see, for example Refs.[12,15]).

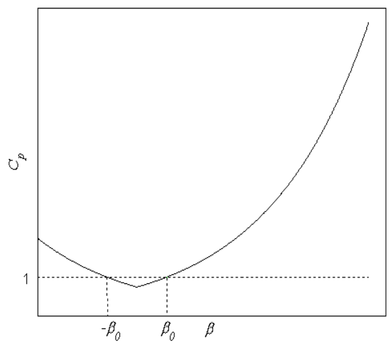


Fig.2 The curve of C_p

In our approach, the feedback gain C_p is designed to have the following form $\begin{cases} C_p = e^{|\beta|-\beta_0} \\ \beta = \dfrac{x_i(n-\tau_i)-\bar{x}_i}{\bar{x}_i} \end{cases}$, where $|\beta|$, called the deviation ratio, represents the extent that the buffer occupancy deviates from the desired point. Let β_0 be the deviation ratio threshold. To show the mathematical property of this parameter, we plot the curve of the function C_p corresponding to some fixed $\bar{x}$ and τ_i in Fig.2. The component C_p now has the property we are looking for due to $\begin{cases} C_p > 1 & |\beta| > \eta \\ C_p = 1 & |\beta| = \eta \\ 0 < C_p < 1 & |\beta| < \eta \end{cases}$. With the increase of deviation ratio, C_p increases correspondingly.

In our algorithm, the node that has the largest β is thought to be the worst one. In the case β is negative, the smaller it is, the lower buffer occupancy is; while in the case β is positive, the larger it is, the worst congestion situation is. Thus β is a good indication of the congestion level of each node without extra calculation. We have the self-tuning controller suggested by

$$\begin{cases} \beta = \max\limits_{i \in direct\ downstream\ nodes} \left(\dfrac{x_i(n-\tau_i)-\bar{x}_i}{\bar{x}_i} \right) \\ C_p = e^{|\beta|-\beta_0} \\ r_0(n+1) = r_0(n) - \alpha \cdot C_p \left[x_{worst}(n-\tau_{worst}) - \bar{x}_{worst} \right] \end{cases} \tag{3}$$

A fine-tuning parameter α ($0 < \alpha < 1$) is added to limit the changing rate of C_p since exponential function often increases too fast when the exponent is large.

The control gain C_P selection can be built on the following stability analyses. To analyze the stability, we have the following Taylor-series expansion

$$\mathrm{e}^{|\beta|-\beta_0} = 1 + (|\beta|-\beta_0) + \frac{1}{2!}(|\beta|-\beta_0)^2 + \ldots + \frac{1}{i!}(|\beta|-\beta_0)^i + \ldots = \sum\nolimits_{i=0}^{\infty} \frac{1}{i!}(|\beta|-\beta_0)^i \tag{4}$$

By substituting (4) into (3), we further have

$$r_0(n+1) = r_0(n) - \alpha \sum\nolimits_{i=0}^{\infty} \frac{1}{i!} \left[\left| \frac{x_{worst}(n-\tau_{worst}) - \bar{x}_{worst}}{\bar{x}_{worst}} \right| - \beta_0 \right]^i \left[x_{worst}(n-\tau_{worst}) - \bar{x}_{worst} \right] \tag{5}$$

Combining (1) with (5), one yields a closed-loop system description. The stability in terms of the buffer occupancy $x(n)$ can be analyzed by using the nonlinear system analysis techniques like those suggested by Ref.[16]. However, we omit this mathematical complexity here. Our selection of the control gain is only following the line of empirical investigation and analysis based on simulations, and the details are to be given in Section 3.

The whole algorithm at the branch point is illustrated in Fig.3. At the center of router control algorithm are two vectors: 1) *multicastVector*, the connection pattern vector where *multicastVector*(i) =1(0) means the ith output port of the router is (not) a downstream branch of the multicast connection and a BCP is (not) expected from the ith downstream branch; 2) *receivedVector*, the responsive branch vector is initialized to $\underline{0}$ and reset to $\underline{0}$ whenever the local transmission rate is updated; while *receivedVector*(i) is set to 1 if a BCP is received from the ith downstream branch. The BCP contains a field named *BO* for filling the buffer occupancy.

At the session establishing time, the desired buffer occupancies of the direct downstream nodes are recorded in *desiredVector* via negotiation option. Upon receiving a data packet or a FCP, the router multicasts it to its output ports specified by *multicastVector*, if corresponding output links are available; otherwise en-queues it in its queue. Upon receiving a feedback BCP from any downstream branch, the router marks its corresponding bit in *receivedVector*, and then select the largest β. If *receivedVector* = *multicastVector*, which implies that all feedbacks have been synchronized, the local transmission rate is then updated. Note that the rate cannot be more than the minimum bandwidth of its output links.

```
At the session establishing period:
Record the desired buffer occupancies of the direct downstream nodes in desiredVector;
On receipt of a data packet or a FCP at the intermediate router:
Multicast data packet based on multicastVector;
On receipt of a BCP from i-th branch:
if multicastVector (i) = 1
{
    receivedVector (i) = 1;
    if (BO – desiredVector (i)) / desiredVector (i) > $\beta$
    {
        $\beta$ = (BO – desiredVector (i)) / desiredVector (i);
        worstDeviation = BO – desiredVector (i);
    }
    if receivedVector = multicastVector
    {
        $C_p = e^{|\beta|-\beta_0}$ ; //uptate its tramsmission rate
        $r = max(0, min(\text{all output links bandwidth}, r - \alpha \cdot C_p \cdot worstDeviation))$;
        receivedVector = $\underline{0}$; // reset receivedVector
        $\beta = -\infty$; // infinitesimal
    }
}
```

Fig.3 Pseudocode of branch point

The feedback consolidation problem is solved naturally in this algorithm. Each BCP just experiences one hop instead of returns to the source node where the feedbacks accumulated at one branch point are very limited compared with those from all the downstream nodes. The small amount of feedbacks will not make the branch point undated.

Rate computation and execution run independently on individual branch point. The distributed processing manner not only simplifies the implementing complexity, but also provides a mechanism to deal with the heterogeneous long propagation delay. This benefit is evident from the following simulation results.

3 Performance Evaluation via Simulation

In this section, we study the performance of the proposed scalable self-tuning controller under a complex network configuration. There are three key performance related issues that merit serious considerations:

(i) The response of the controller from the initial state to stead state, i.e., the duration it takes for system to reach an equilibrium state;

(ii) The steady state of the bottleneck, of particular interest is the buffer occupancy;

(iii) The average link utilization of the bottleneck link.

Our designed simulation topology is shown is Fig.4. The multicast source and intermediate points are represented by Node1~Node13. Links are denoted by L_1~L_{19} with the corresponding forward/backward delay in the bracket measured in T (T=1ms). Node1 is the multicast session source; Node4, Node5, Node6, Node9, Node10, Node12 and Node13 are the destinations. All links have bandwidth 300Mb/s except L_3 and L_{13} with bandwidth 75Mb/s. The configuration includes the cases of node having single-branch (e.g. Node11), nodes having multiple

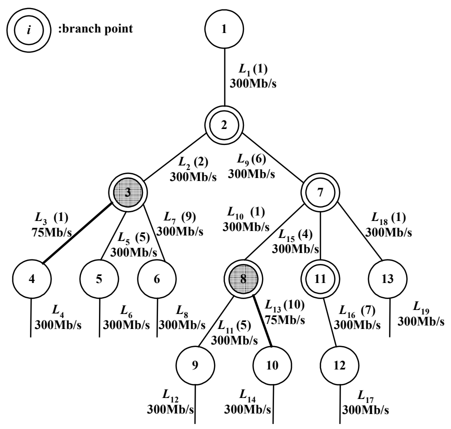
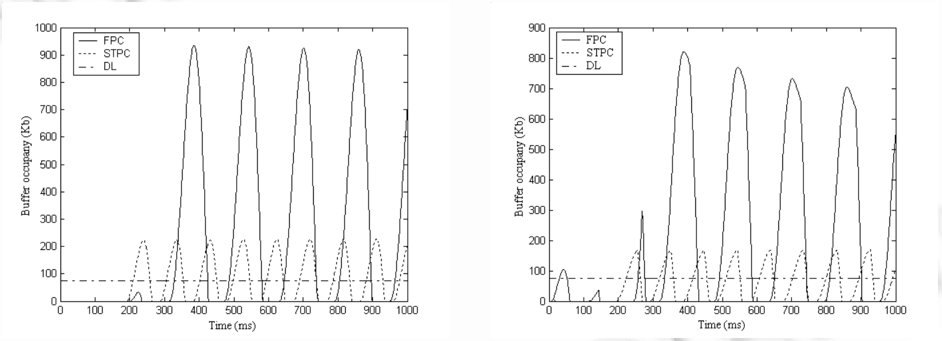


Fig.4 Multicast simulation topology

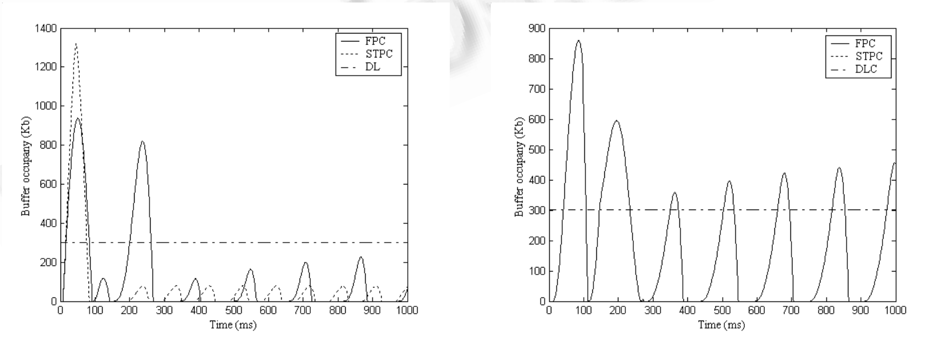
branches (e.g. Node3 and Node7), nodes having different hops to the source, heterogeneous RTTs and various link bandwidths. It is believed that this model is representative enough for the purpose of studying our proposed scheme in a WAN environment. We carry out extensive simulations using the software MATLAB, and compare the self-tuning proportional control (STPC) with the fixed proportional control (FPC) described by Eqs.(1) and (2). FPC is named for its invariable proportional gain once the control begins.

The network configuration that we investigate into involves two bottleneck links, L_3 and L_{13}, which bring Node3 and Node8 into being the bottlenecks. The desired buffer occupancy is set proportionally to its minimum outgoing link capacity to reflect the individual transmission capacity as follows: $\bar{x}_3 = \bar{x}_8 = 75\text{Kb}$, $\bar{x}_4 = \bar{x}_5 = \bar{x}_6 = \bar{x}_7 = \bar{x}_9 = \bar{x}_{10} = \bar{x}_{11} = \bar{x}_{12} = \bar{x}_{13} = 300\text{Kb}$. The feedback gain C_p used in FPC is set to be 0.002; the other parameters used in STPC are chosen as $r_1(0)$=42Mb/s, α = 0.001, β_0 = 0.1, MDR = 0, PDR = 1000Mb/s. The simulation duration is 1000ms and the results are shown in Fig.5 ~ Fig.8. The notation DL in the figures represents the desired level.



(a) Node3 (b) Node8

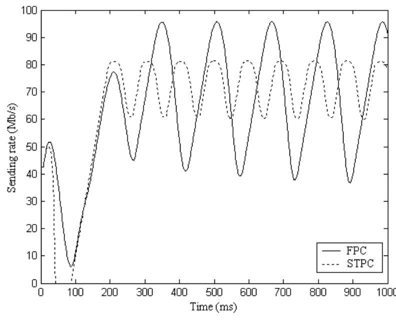
Fig.5 Bottleneck buffer occupancy



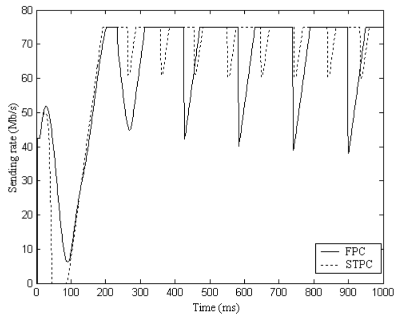
(a) Node7 (b) Node11

Fig. 6 Nonbottleneck buffer occupancy

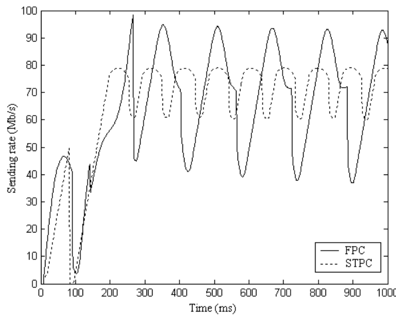
From Figs.5(a) and (b), it is found that the buffer occupancy of the bottleneck nodes in STPC is much smaller than that in FPC, and nearer to the desired buffer level. Take Node3 for an example, the maximal buffer occupancy in FPC is 920Kb while 227Kb in STPC, almost one fourth of the former. High occupancy results in a long queuing time due to extra queue build-up, which would degrade the responsiveness of the feedback and subsequently the efficiency of the control scheme. Figures 6(a) and (b) depict the non-zero buffer occupancies of the non-bottleneck nodes. STPC still shows superiority in smaller buffer occupancy. Although it fluctuates around the desired level, its smooth change shows that it is still acceptable.



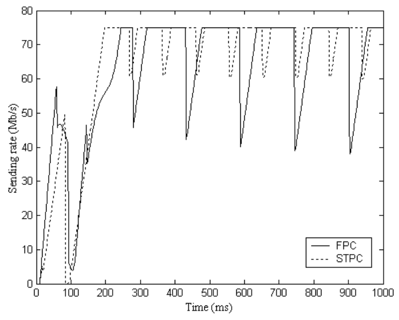
(a) Node1 (source)



(b) Node3

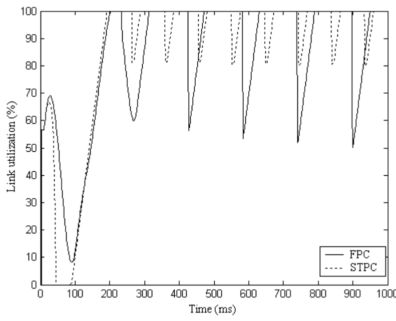


(c) Node7

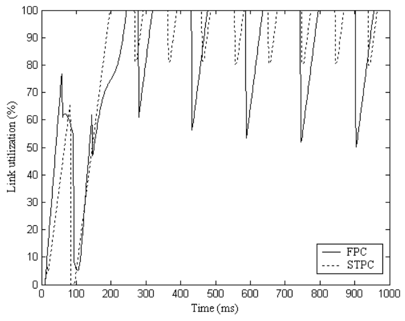


(d) Node8

Fig.7 Transmission rates of the source and branch points



(a) L_3



(b) L_{13}

Fig.8 Bottleneck link utilization

The transmission rates of the source and branch points are shown in Fig.7. In both schemes the rates fluctuate around the bottleneck link capacity 75Mb/s, but the swing is smaller in STPC, meaning a better steady state performance. Furthermore, by more closely looking into Fig.7, one notes STPC also demonstrates better transient dynamics. STPC yields a shorter response time than FPC. In STPC, it takes about 160ms for the source node to come into a regular pattern; while in FPC, that is about 260ms.

From Fig.8, one observes there is certain advantage of STPC over FPC, which is that STPC achieves a better bottleneck link utilization. The average utilization of L_3 in FPC is 82.93%, while in 85.93% STPC; the average utilization of L_{13} in FPC is 82.43%, while 85.43% in STPC.

In summary, the simulation results show that under our proposed STPC scheme, the response time is shorter and the buffer occupancy fluctuates smoothly around the desired point. The utilization of the bottleneck link is excellent.

4 Conclusions

This paper presents a distributed congestion control method in multicast communication networks. It uses a self-tuning proportional control to regulate the transmission rate of not only the source but also the intermediate nodes. The proportional control gain in this scheme is shown to be able to adjust automatically depending on the network load. Simulation results demonstrate that this method can achieve good system dynamics along with excellent link utilization. It is able to scale to a large number of receivers and to be implemented in a heterogeneous environment with different link capacities and delays. Our further research would investigate into the TCP-friendly related issues in multicast congestion control along this line of study.

References:

[1] Deering S. Host extensions for IP multicasting. RFC1112, 1989.

[2] Benmohamed L, Meekov SM. Feedback control of congestion in packet switching networks: The case of a single congested node. IEEE/ACM Trans. on Networking, 1993,1(6):693~708.

[3] Keshav S. A control-theoretic approach to flow control. In: Proc. of the ACM SIGCOMM'91. Zurich: ACM Press, 1991. 3~15.

[4] Benmohamed L, Meerkov SM. Feedback control of congestion in packet-switching networks: The case of multiple congested nodes. Int'l Journal of Communication Systems, 1997,10(5):227~246.

[5] Tzeng HY, Siu KY. On max-min fair congestion control for multicast ABR services in ATM. IEEE Journal on Selected Areas in Communications, 1997,15(3):545~556.

[6] Saito H, Kawashima K, Kitazume H, Koike A, Ishizuka M, Abe A. Performance issues in public ABR service. IEEE Communications Magazine, 1996,(11):40~48.

[7] Zhang X, Shin KG. Statistical analysis of feedback synchronization signaling delay for multicast flow control. In: Proc. of the IEEE INFOCOM 2001. Anchorage, 2001. 152~1161.

[8] Cho YZ, Lee SM, Lee MY. An efficient rate-based algorithm for point-to-multipoint ABR service. In: Proc. of the IEEE GLOBECOM 1997. Phoenix, 1997. 790~795.

[9] DeLucia D, Obraczka K. Multicast feedback suppression using representatives. In: Proc. of the IEEE INFOCOM 1997. Kobe, 1997. 463~470.

[10] Rizzo L. PGMMCC: A TCP-friendly single-rate multicast congestion control scheme. In: Proc. of the ACM SIGCOMM 2000. Stockholm, 2000. 17~28.

[11] Lee SH, Lim JT. Multicast ABR service in ATM networks using a fuzzy-logic-based consolidation algorithm. IEE Proceedings – Communication, 2001,148(1):8~13.

[12] Zhang X, Shin KG, Saha D, Kandlur DD. Scalable flow control for multicast ABR services in ATM networks. IEEE/ACM Trans. on Networking, 2002,10(1):67~85.

[13] Shi S, Waldvogel M. A rate-based end-to-end multicast congestion control protocol. In: Proc. of the 5th IEEE Symp. on Computers and Communications. Antibes, 2000. 678~686.

[14] Crowcroft J, Paliwoda K. A multicast transport protocol. In: Proc. of the ACM SIGCOMM. Stanford: ACM Press, 1988. 247~256.

[15] Golestani SJ, Sabnani KK. Fundamental observations on multicast congestion control in the Internet. In: Proc. of the IEEE INFOCOM 1999. New York, 1999. 990~1000.

[16] Vidyasagar M. Nonliner Systems Analysis. 2nd, Philadelphia: Society for Industrial and Applied Mathematics (SIAM), 2002.