ELSEVIER

Computer Networks 38 (2002) 645–662

COMPUTER NETWORKS

www.elsevier.com/locate/comnet

An efficient traffic control scheme for TCP over ATM GFR services

Chia-Tai Chan[a], Pi-Chung Wang[a,b,*], Yaw-Chung Chen[b]

[a] *Telecommunication Laboratories, Chunghwa Telecom Co. Ltd., Taipei 106, Taiwan, ROC*

[b] *Department of Computer Science and Information Engineering, National Chiao Tung University, Hsinchu 30050, Taiwan, ROC*

Received 8 February 2000; received in revised form 13 June 2000; accepted 5 September 2001

Responsible Editor: J. Sterbenz

Abstract

In ATM networks, the guaranteed frame rate (GFR) service category has been defined to support user applications which are neither able to specify the range of traffic parameter values, nor able to comply with the behavior rules. It provides a packet-level rate guarantee with a given maximum packet size. The service specifies that the excessive traffic from a user should share the available resource fairly. In this paper, we investigate TCP/IP packet transmission over ATM by using a selective packet-discard strategy with tracking of the available buffer space and a packet push-out buffering scheme to accommodate the GFR service. The simulation results show that our proposed method fulfills the requirements of GFR service as well as improves the TCP throughput under the common FIFO scheduling. A feasible implementation approach is also addressed.

Keywords: ATM networks; Guaranteed frame rate; Selective packet-discard; Packet push-out buffering

1. Introduction

The guaranteed frame rate (GFR, formerly called UBR+) service category has been proposed as an enhancement to the ATM UBR (unspecified bit rate) service that guarantees a minimum throughput for non-real-time applications at the frame level. This service category is intended for user applications which are neither able to specify traffic parameter values, nor able to comply with the behavior rules required by existing ATM services [1–5]. The GFR service needs only minimal interactions between users and ATM networks. In addition, the service also specifies that the excessive traffic of each user should fairly share the available resource. As a result, designing an efficient buffer management scheme for supporting the GFR service is a key issue towards the successful deployment of GFR.

The effective throughput or goodput is defined as the throughput that is valid in terms of high-layer protocols. The goodput in terms of the higher-layer protocol over ATM networks may be quite low due to the wasted bandwidth for transmitting cells of corrupted packets, which can be caused by the

* Corresponding author. Tel.: +886-3-5731851; fax: +886-3-5727842.

E-mail address: pcwang@csie.nctu.edu.tw (P.-C. Wang).

1389-1286/01/$ - see front matter
PII: S1389-1286(01)00278-X

packet fragmentation. Here we assume that the cell transmission is error free. It has been demonstrated that the selective packet-discard strategies, i.e., early packet discard (EPD) and partial packet discard (PPD), alleviate the packet fragmentation problem and restore goodput [8].

In this study, we propose a traffic control scheme by using a selective packet-discard strategy with tracking of the available buffer space (ABS), and a packet push-out buffering scheme to support the quality of GFR service using a FIFO queuing discipline. The upper-layer protocol discussed here is TCP/IP. The merit of the proposed GFR traffic control approach is the improvement in both transmission efficiency and fair sharing of network resources, with feasibility of implementation. The organization of this paper is as follows: Section 2 gives an overview of TCP flow-control behavior, its related packet-discard strategies and several GFR implementation alternatives. In Section 3, we present the functional approach of our proposed scheme. The system model and simulation results are discussed in Section 4. Section 5 concludes the work.

2. Overview

2.1. TCP flow-control behavior

TCP uses a window-based flow-control mechanism; its window-adjustment algorithm consists of two phases. A connection begins with the slow-start phase. When a new connection is established, its congestion window (CWND) is initialized to one segment. Upon receiving an ACK packet, the CWND is increased by one segment; this process continues until it reaches a slow-start threshold (SSTHRESH, typically 65,535 bytes). The sender can transmit up to either CWND or receiver's advertised window, whichever smaller will be chosen. It can be shown that CWND actually increases exponentially every round-trip time. When congestion occurs (indicated by a timeout or by reception of duplicate ACKs), one half of the current window-size value (the smaller value between the CWND and the receiver's advertised window, with a minimum of two segments) is saved in SSTHRESH. Additionally, if the congestion is indicated by an expired timer, the CWND will be set to one segment. If CWND were no greater than SSTHRESH, TCP is in slow-start phase; otherwise it is in congestion avoidance phase. In the latter case, CWND is increased by ((segment size $\times$ segment size)/CWND) each time an ACK is received, which results in a linear increase of CWND. TCP Reno implements the fast retransmit and recovery algorithms [6] that allow the connection to quickly recover from isolated segment losses.

It is known that fast retransmit and recovery cannot recover multiple packet losses, it only causes the exponential increment phase to last a very short time, and the linear increment phase to begin with a very small window. As a result, TCP operates at a very low rate and loses a certain amount of throughput. TCP new Reno is a modification to fast retransmit and recovery. In TCP new Reno, the sender can recover from multiple packet losses without having to timeout [7].

2.2. Related packet-discard strategies

An AAL-5 encoded TCP/IP data packet, with 1 or more bytes of TCP payload, cannot fit in a single ATM cell. The destination cannot reassemble the corrupted packet with any cell missing. In order to provide reliable and transparent data transport service, the source must be requested to retransmit the entire corrupted packet. Thus, whenever one of the cells constituting a packet is lost, all followed cell transmissions of that packet are certainly unnecessary. Therefore, the packet fragmentation may result in low goodput. The packet-discard strategies EPD and PPD are able to accommodate the packet fragmentation problem and restore goodput. In PPD, if a cell is dropped from a switch buffer, the subsequent cells belonging to the same higher-layer protocol data unit (e.g., TCP/IP packets) are discarded. It can be shown that PPD improves performance to a certain level, but its goodput still needs improvement. Therefore, EPD has been proposed, in which the entire higher-layer protocol data unit is dropped when the switch buffer occupancy reaches a predefined threshold. In Ref. [8], the authors have

shown that the goodput of EPD is better than PPD.

The setting of the EPD threshold determines how efficiently the buffer can be used and how often the cell dropping may occur. In an analysis based on the worst-case assumption [9], the EPD scheme that ensures 100% goodput under overload situation requires an extra buffer space of one maximum packet length beyond the EPD threshold per active virtual connection. In Ref. [10], the switch sets up EPD threshold according to the number of active virtual circuits, and 100% goodput can be achieved with substantially smaller buffer space than that predicted by the worst-case analysis. Obviously, it improves the buffer utilization and the TCP throughput.

2.3. A feasible realization of GFR

As described in Section 1, the GFR service has been defined to provide traffic streams with a minimum packet rate guarantee for packets that do not exceed a maximum packet size. In addition, the service also specifies that the excess traffic of each user should fairly share the available resource. An efficient GFR service scheduling strategy should allow each flow passing through a network node to get a fair resource share [11]. A fair queuing scheduler serves flows in proportion to certain predetermined shares, as well as protecting from the interference of ill-behaved sources. Several fair queuing service disciplines have been discussed extensively in the literature. Examples are weighted fair queuing [12], virtual clock [13], packet-by-packet generalized processor sharing [14,15], and self-clocked fair queuing [16,17]. However, the drawbacks of these dynamic time-priority schemes are the high processing overhead for tracking the progress of tasks and scheduling the time-stamped packets. It has been suggested in Ref. [4] that a simple rate-guaranteeing discipline (i.e., weighted round robin) with per-VC queuing is indeed necessary to ensure GFR service. However, the per-VC queuing would greatly complicate the switching system design. In Section 3, we propose a control approach that uses FIFO queuing in network switching nodes to support the GFR service requirements. We demonstrate that it is possible to use FIFO queuing instead of per-VC queuing to fulfill the requirements of GFR service.

3. Proposed GFR traffic control

GFR can be used between ATM edge devices. For example, IP routers connected through an ATM network can set up GFR VCs between them for data transfer. Fig. 1 illustrates such a case, in

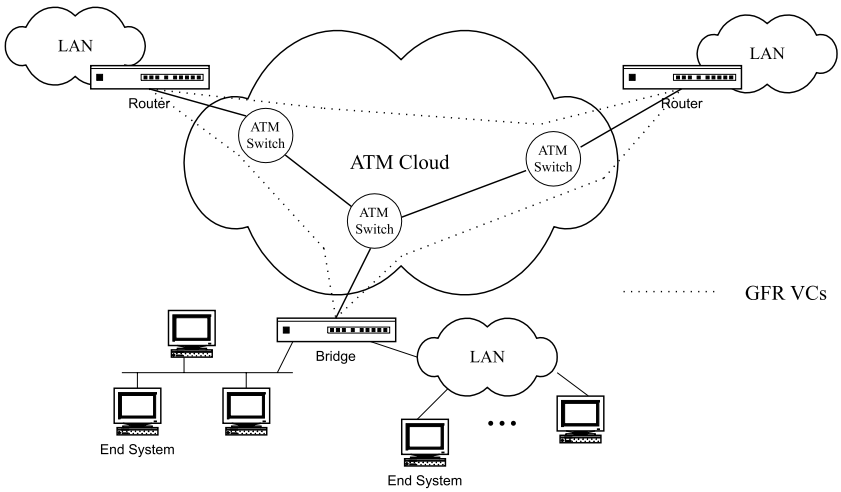


Fig. 1. GFR services in IP/ATM internetworks.

which the ATM cloud connects LANs and routers. ATM end systems may also establish GFR VC connections between one another to obtain a minimum throughput guarantee. In ATM networks, the GFR traffic usually coexists with other higher priority CBR/VBR traffic. Consequently, the available bandwidth allocated to GFR VCs varies dynamically. However, it is expected that dedicated bandwidth should be allocated for GFR services to avoid the service starvation. The GFR traffic should be served in the manner, such as class based weighted round robin, to achieve the configured service rate.

To provide guaranteed GFR services, we propose a packet discard and push-out buffering approach, which consists of a selective packet-discard mechanism with tracking of the ABS, and a packet push-out buffering scheme. The proposed buffering scheme ensures the fair sharing of network resources and avoids the misbehaved connection from occupying excessive buffer space. From the previous discussion, the selective EPD strategy could achieve 100% goodput that alleviates fragmentation problem. To further improve the TCP throughput, we introduce a selective packet-discard strategy with ABS tracking. Our scheme not only fulfills the GFR service requirements, but also achieves 100% goodput that provides a nearly optimal TCP throughput.

3.1. Selective packet-discard strategy with tracking of available buffer space

Given an ATM switch, we assume that an output port has a FIFO queue of size Q (in cell units) allocated for n GFR connections; also we assume that the maximum packet length is M cells.

The EPD strategy is based on either a static threshold or dynamic threshold. In the latter, the switch sets the threshold according to the number of active virtual circuits. Our proposed method also uses a dynamic threshold based on the ABS. The switch maintains a state variable B to estimate the ABS, and a state variable L_i to count the incoming packet length for connection i. The proposed method operates as follows:

1. Initially, B is set to Q, and L_i is set to zero $(1 \leqslant i \leqslant n)$.
2. If B is no less than one, the first cell of an arrived packet will be admitted into the buffer, and B will be set to $B - M$. Otherwise, the switch will drop the first arriving cell and all subsequent cells belonging to the same packet. Note that if the incoming line is pumping cells into the buffer faster than the outgoing line of the switch, it still causes packet partial dropped. Therefore, if the arriving cell causes buffer overflow then the push-out process will be activated. The detail discussion of push-out process will be illustrated in the later.
3. Whenever a cell is transmitted, B is incremented by one. When the first cell of a packet from connection i is admitted into the buffer, the L_i is incremented by one, and the switch starts to count the length of this incoming packet until its last cell is received, then B is updated to $B + M - L_i$.

There may be multiple cells arriving within one cell-slot time; under such situation, the cell service sequence will follow the predetermined order. The switch can identify the last cell of the incoming packet by checking the ATM-layer user-to-user indication bit specified in the cell header. Once a switch discards the first cell of an incoming packet, it continues to discard the subsequent cells of that packet.

By tracking the ABS, the packet-discard method with dynamic threshold effectively improves the buffer utilization. As a result, our proposed method can achieve 100% goodput. Moreover, it improves the TCP throughput to a nearly optimal level.

3.2. Packet push-out buffering scheme

The design of simple and efficient buffer management approaches for accommodating GFR service requirements is an important issue toward the successful development of GFR. As described in Section 2.3, several sophisticated scheduling disciplines and the per-VC queuing would greatly complicate switching system design, such as large separate queue structures, and scheduler state. In contrast, a FIFO service discipline is easy to implement and requires very little scheduler state. However, it has no protection between

well-behaved and misbehaved connections. It needs a specific buffering scheme to police the incoming packets to fulfill the GFR service requirements. A flow can acquire a greater service share by sending more traffic to keep a higher occupancy in the FIFO queue. It is thus important to fairly allocate the network resources even in the presence of ill-behaved sources under heavy-load conditions. One possible solution is either to serve or to discard the arriving packets according to the fairness policy. The fairness of GFR services is defined that excess traffic from each GFR connection should share available resources fairly. The guaranteed minimum packet rate is first allocated for each VC, then the rest of the available bandwidth is shared equally (i.e., minimum packet rate plus equal sharing, MPR + ES). There are other definitions of fairness, such as weighted allocation, which assigns available bandwidth to all active connections according to their relative performance demands.

A switch can use per-VC accounting to realize the dynamic sharing of the buffer space among all TCP flows. Let W be the current bandwidth available for GFR traffic in an output port, Q be the buffer size in cells and r_i be the requested service rate of VC_i, where $1 \leqslant i \leqslant n$. In a FIFO service discipline, the service rate r_i can be achieved if VC_i keeps an average buffer occupancy of b_i cells in the FIFO queue, where

$$\frac{r_i}{W} \leqslant \frac{b_i}{Q} \leqslant \frac{b_i}{\sum_{j=1}^{n} b_j} \quad (1 \leqslant i \leqslant n).$$

Then, VC_i can obtain a service rate $W(b_i/\sum b_j)$ which is no less than r_i. That means the throughput experienced by a connection VC_i is proportional to its average fraction of buffer occupancy, which can be preset to a threshold TL_i. Hence, if we keep the buffer occupancy of VC_i at a desired level, its service rate can be controlled. In this work, we adopt a simple MPR+ES method, and assign an appropriate TL_i for each VC_i as follows:

$$TL_i = \left[MPR_i + \left(W - \sum_{j=1}^{n} MPR_j \right) w_i \right] \frac{Q}{W},$$

$$\text{where } w_i = \frac{MPR_i}{\sum_{j=1}^{n} MPR_j},$$

MPR_i is the guaranteed minimum packet rate of VC_i and n is the total number of GFR connections.

Let NC_i be the actual number of cells for VC_i in the buffer. Our packet push-out buffering scheme operates as follows:

1. When a first packet cell of VC_i arrives, if $B > 0$ the whole incoming packet will be admitted into the FIFO. If $B \leqslant 0$ and $NC_i + M$ is no greater than its threshold TL_i, the push-out process will be activated. The push-out mechanism will also be activated for any incoming cell, excluding the first, that causes buffer overflow.
2. The queue manager selects a VC_j which has the maximum value of $NC_j - TL_j$ $(1 \leqslant i, j \leqslant n,\ i \neq j)$ and pushes out its last packet. Then, the length of the pushed-out packet is added to B. Note that the queue manager should push-out packets continuously until $B > 0$. Then, according to our approach, it may find ABS for the incoming packet of VC_i. It is possible that the queue manager will push out a packet which has not yet completely arrived. Therefore, the queue manager will need to update state information so that cells from the push-out packet that have not yet arrived at the buffer will be discarded when they do arrive. The detailed implementation will be addressed in next section.
3. Otherwise, all cells of the incoming packet will be discarded.

The functional diagrams of cell receiving and transmission for our proposed control approach is illustrated in Figs. 2 and 3, respectively.

It is true that smaller buffer size reduces goodput for TCP traffic over ATM. Since the proposed strategy is packet-based discard and push-out buffering scheme, it will cause certain problems that can be referred as a *goodput beat down* problem, explained as follows. If the FIFO queue size is much smaller than nM (i.e., $Q \ll nM$, where n is the number of VCs passing though the buffer) then most packets will experience partial discard. In some circumstances, it is likely that sources will be prevented from getting their effective throughput. To provide a certain level of goodput, a minimum buffer size is required. The proper buffer size

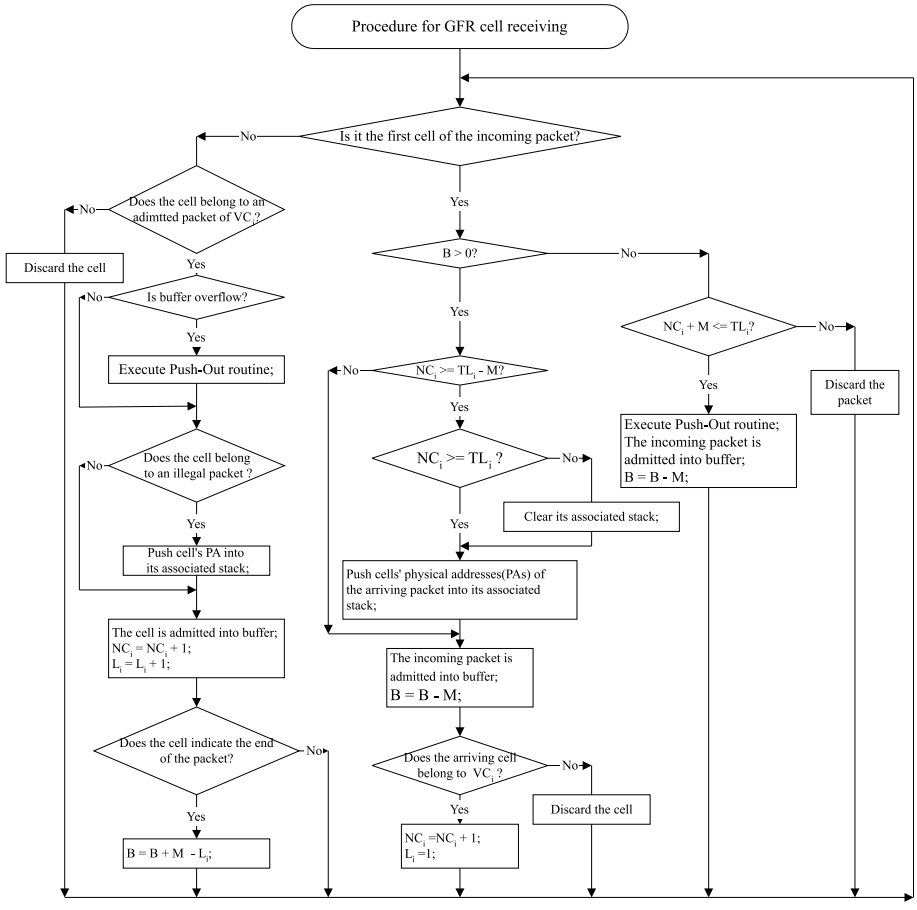


Fig. 2. The procedure of cell receiving in proposed control approach.

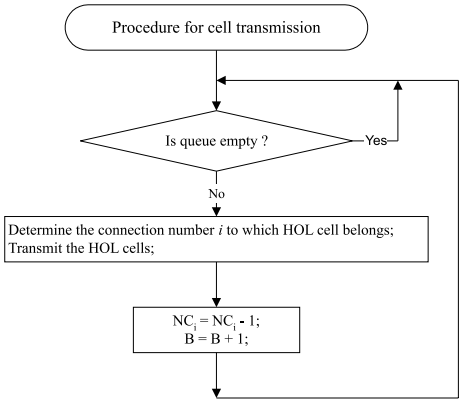


Fig. 3. The procedure of cell transmission in proposed control approach.

should be proportional to the number of VCs and the packet size.

A switch uses per-VC accounting to accomplish the dynamic sharing of the buffer space among all TCP flows. Cells belonging to the same VC are logically organized in a double-linked list. To ensure a fair share of the available resource, we use push-out scheme to prevent the misbehaved connections from occupying excessive space in the FIFO queue. Then, a simple FIFO service discipline can fulfills the requirements of GFR services.

3.3. Realization of the proposed traffic control method

We have mentioned that our proposed control approach consists of a selective packet-discard

strategy and a packet push-out buffering scheme, which can be realized with an architecture consisting of four major functional components: a packet-discard controller (PDC), a cell dispatcher (CD), a stack controller (SC) and a push-out controller (POC), as illustrated in Fig. 4. Packets from the ith virtual connection with an assigned threshold TL_i are defined as legal if the number of cells from the same connection in the FIFO queue is no greater than TL_i.

The PDC discards packets when there is no ABS, and manipulates the control variables NC_i $(0 \leqslant i \leqslant n)$ and B. When a first packet cell arrives, it will be accepted by PDC if there is ABS for the whole packet (i.e., $B \geqslant 1$). Otherwise, the packet will be discarded. The control variable NC_i is used to turn on/off the POC and SC for the ith virtual connection. Once the first cell of an incoming packet arrives and is admitted into the buffer, if $NC_i \geqslant TL_i - M$ then all cells' physical addresses (PAs) of the arriving packet must be stored into the associated stack for possible push-out operation in the future, because it may be an illegal packet. However, an illegal packet may become legal later on. Therefore, if $TL_i > NC_i \geqslant TL_i - M$ the PDC immediately activates the SC to clear the associated stack before the first cell's PA of the incoming packet is inserted into it. This is because that cells' PAs in the stack become legal and thus their associated cells cannot be discarded. If $NC_i \geqslant TL_i$, the PDC will activate the SC to push the cell's PA into its associated stack. The procedure for maintaining the stacks of PDC is illustrated in Fig. 5.

An arriving cell will be classified by the PDC, then written into the cell pool. Prior to this process, the associated stack number is extracted by the CD and its PA is stored into the POC. The POC consists of several modules, through which PAs are stored. Once the POC receives a push-out signal, it activates the SC to pop out illegal packets

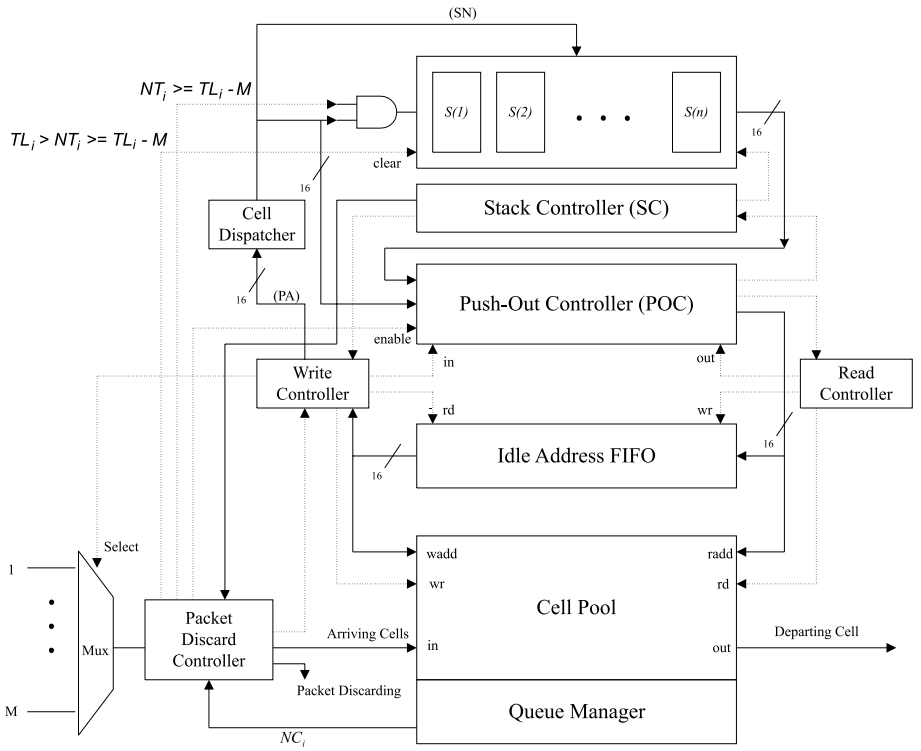


Fig. 4. Implementation architecture of the proposed traffic controller.

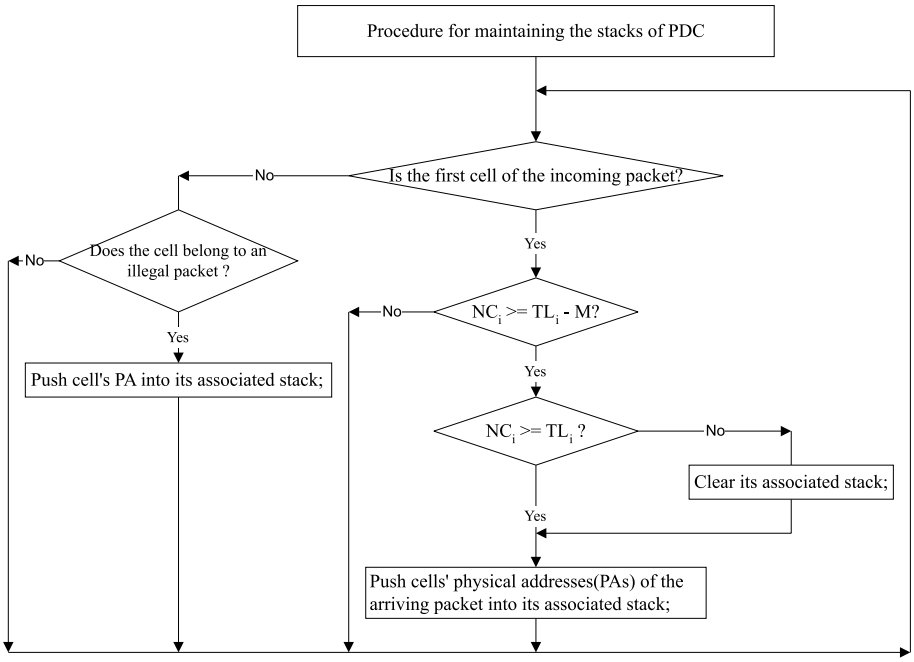


Fig. 5. The procedure of maintaining the stacks in PDC.

from the stack with associated connection occupying excessive buffer space. The selected candidate will be the VC that has the largest number of illegal packets in the associated stack. Each time the push-out routine is activated, illegal packets will be popped out continuously until the number of popped cells is greater than M, then the cell PAs of the popped packet will be returned to POC.

The SC consists of a set of stacks, each of which is associated with a connection, but not all connections need to use a stack. The purpose of the SC is to keep the illegal packets of a VC in the associated stack for possible push-out process. Since the selective discard strategy is packet based, an indication bit is used to identify the last cell's PA of a packet. As a result, once the SC pops the last cell of a packet it continues to pop out remaining cells of the same packet. The information regarding the popped-out packet (i.e., the number of popped-out cells) must be sent to the PDC for the sake of maintaining an accurate NC value. In addition, it is possible that the popped-out packet is not complete, because part of the packet cells might have not arrived yet, thus the SC must inform the PDC to discard the subsequent cells of the popped-out packet and keep the value of B correct.

The service discipline of POC is simply FIFO, in which the PA of a cell is stored in a 16-bit register. Once the new PA of an incoming cell is to be inserted into POC, the controller just appends the new PA to the FIFO modules. When a head-of-line cell is scheduled for transmission, its PA is retrieved from POC to identify the cell location in the cell pool. Once a cell has been transmitted, the content of all 16-bit registers will be shifted one position to the right; hence the PA in the POC will be overwritten automatically. In the mean time the transmitted cell's PA will be written into the idle-address pool for future incoming cells. The detailed realization can be found in Ref. [18].

4. Simulation and performance evaluation

4.1. Simulation environment for small network configuration

Three network configurations are chosen to illustrate the effectiveness and scalability of our

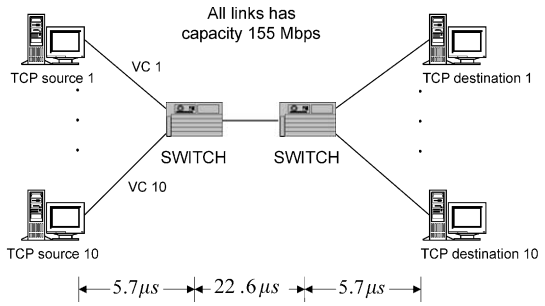


Fig. 6. A TCP configuration of 10 sources.

proposed approach. We evaluate the small network configuration first, as shown in Fig. 6. Simulations are performed based on this configuration for new Reno TCP connections with 10 sources, given that the link transmission rate is 155 Mbps. The cell transmission time is about 2.78 μs, which is defined as a cell-slot time. The propagation delay is two slot times between the host and the switch, and eight slot times between switches. Ten TCP connections complete for network resources (i.e., buffer space, bandwidth) simultaneously and 15.5 Mbps is allocated for each connection. Assume that the 10th TCP source is greedy, and it sends packets at almost three times (i.e., 45 Mbps) of the allowed rate. The simulation time was 54 s which is much longer than any round-trip delay in real applications; this is sufficient for the simulation process to reach the steady state as well as to collect the necessary information. The amount of transferred data is more than 1.2 Gb during the simulation period. The maximum TCP segment size is 512 bytes, common in IP networks. A buffer size of 200–1000 cells per port is appropriate for a small switch.

Under the FIFO service discipline, we investigate and compare the performance characteristics of our proposed strategy with EPD and EPD + push-out strategies. The EPD + push-out strategy is a combination of partial buffer sharing (PBS) with the well-known push-out scheme, in which an arriving packet is admitted only if the queue length is less than the EPD threshold. In addition to this, if the queue occupancy exceeds the EPD threshold and $NC_i + M$ is no greater than its threshold TL_i, it will activate the push-out process.

The major performance measure considered here are the goodput, the TCP throughput and the fair sharing of the network resource. The following section explains the results of the proposed traffic control scheme.

4.2. Numerical results for small network configuration

4.2.1. Total goodput versus EPD threshold

The relationship between the total goodput and the buffer size beyond the EPD threshold is presented here. The buffer size in excess of the EPD threshold varies from 1 to 5 segments. From the results shown in Fig. 7, the proposed traffic control method always experiences a 100% goodput. The major reason is that the selective packet-discard strategy tracks the ABS for each incoming packet, which will be admitted into the buffer only when there is enough buffer space. In addition, the push-out scheme is also a packet-based mechanism. Therefore, none of the partially received packet will occur in our control approach. Obviously, the proposed control scheme can optimize the goodput.

On the other hand, in EPD and EPD + push-out schemes the total goodput increases gradually as the buffer size in excess of the EPD threshold increases, as illustrated in Fig. 7. When the buffer size is equal to n times the maximum packet size, where n is the number of connections, EPD can achieve 100% goodput. Due to the effect of push-out process, the EPD + push-out scheme achieves higher goodput than EPD scheme. The push-out process has little effect under the light traffic load, but it does improve the goodput under the heavy traffic condition. This is because a partially transmitted packet in the EPD control scheme may be transmitted completely with the EPD + push-out control, which reduces the number of partially received packets. The major difference between EPD + push-out and the proposed approach is the setting of the EPD threshold. Our proposed method uses a dynamic threshold based on the ABS. By tracking the ABS, the packet-discard method with dynamic threshold effectively improves the buffer utilization. The degree of throughput improvement may depend on the buffer size in excess of the EPD threshold.

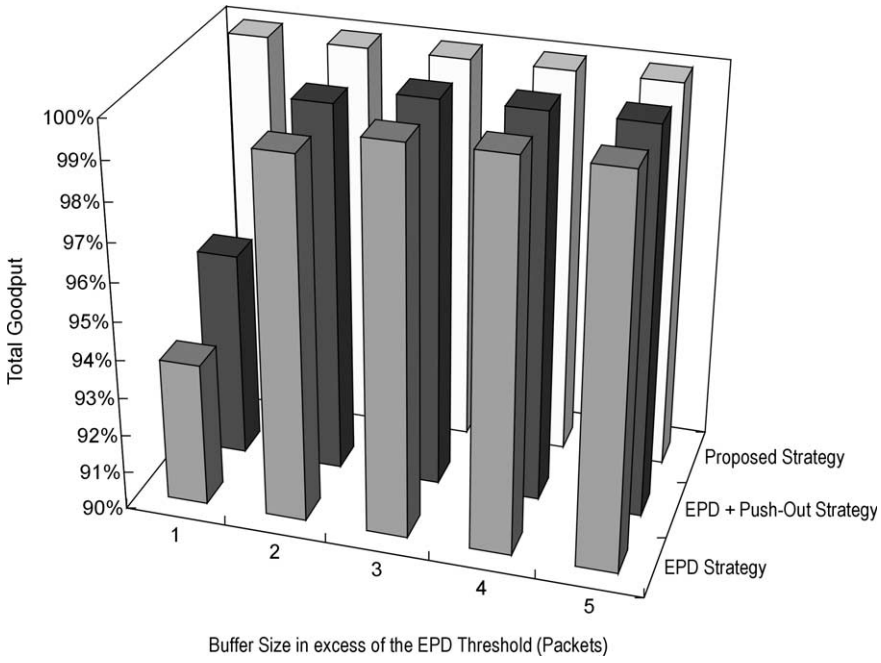


Fig. 7. The goodput versus the buffer size in excess of the EPD threshold.

4.2.2. Total goodput versus buffer size

The relationship between the total goodput and the buffer size is presented in Fig. 8. The total buffer size varies from 200 to 1000 cells. The buffer size in excess of the EPD threshold is two segments. Although the total goodput increases as the buffer size increases in EPD and EPD + push-out schemes, the total goodput of the proposed approach is even better, it is always maintained at 100% regardless of the change of buffer size.

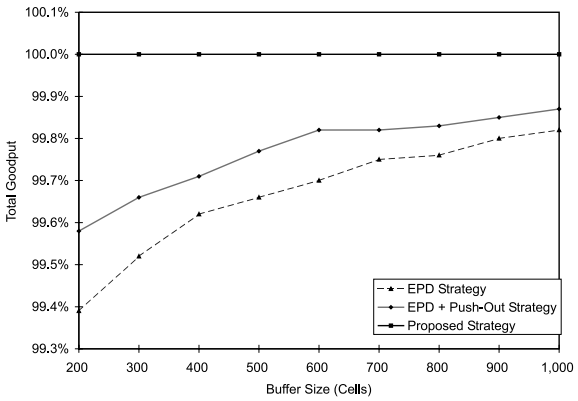


Fig. 8. The goodput versus the buffer size.

4.2.3. TCP throughput versus EPD threshold

Fig. 9 shows the relationship between the total TCP throughput and the buffer size in excess of the EPD threshold. The total buffer size is 200 cells, and the buffer size in excess of the EPD threshold varies from 1 to 10 segments. Since increasing buffer utilization improves the total TCP throughput, the selective packet-discard strategy will drop a packet only if there is not enough buffer space. In EPD or EPD + push-out strategy, it is possible that the buffer space is available but the incoming packet is still discarded because the buffer occupancy exceeds the EPD threshold. Through tracking the ABS, it can achieve much higher buffer utilization. Obviously, our proposed strategy provides higher TCP throughput than EPD strategy, as shown in Fig. 9. Furthermore, we have also observed that in EPD and EPD + push-out schemes, the total TCP throughput decreases gradually as the buffer size in excess of the EPD threshold increases for more than two segments. This is because the increase of the

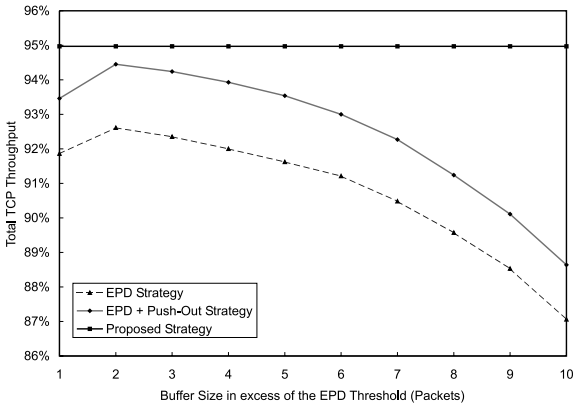


Fig. 9. The TCP throughput versus the buffer size in excess of the EPD threshold.

buffer size in excess of the EPD threshold will lessen the available buffer size to reduce, and it affects the loss probability of the incoming packets.

4.2.4. TCP throughput versus buffer size

With FIFO queuing, the data flow of a misbehaved TCP source may always occupy a large portion of the buffer space. In EPD, when congestion occurs, even the well-behaved sources may suffer a large number of packet retransmission. By using our control approach, the number of packet retransmission is reduced by ~35–40% for well-behaved sources with respect to EPD and ~10–12% with respect to EPD + push-out. Moreover, the number of packet retransmissions for misbehaved sources in our scheme is almost same as that of EPD. This result is the effect of combining the selective packet-discard strategy with packet push-out buffering scheme. It is obvious that the proposed strategy improves the total TCP throughput because it reduces the total number of retransmitted packets, as shown in Fig. 10.

The relationship between the TCP throughput and the buffer size is presented in Fig. 11. The total buffer size varies from 200 to 2000 cells. The buffer size in excess of the EPD threshold is two segments. Fig. 11 shows that the larger the buffer size, the higher the TCP throughput. When the buffer size is set to 2000 (cells), the total TCP throughput of the proposed strategy is 96.8%, while it is 92.8% and 96.4% for EPD and EPD + push-out strategy, respectively. Clearly, the total TCP throughput is improved in our proposed approach.

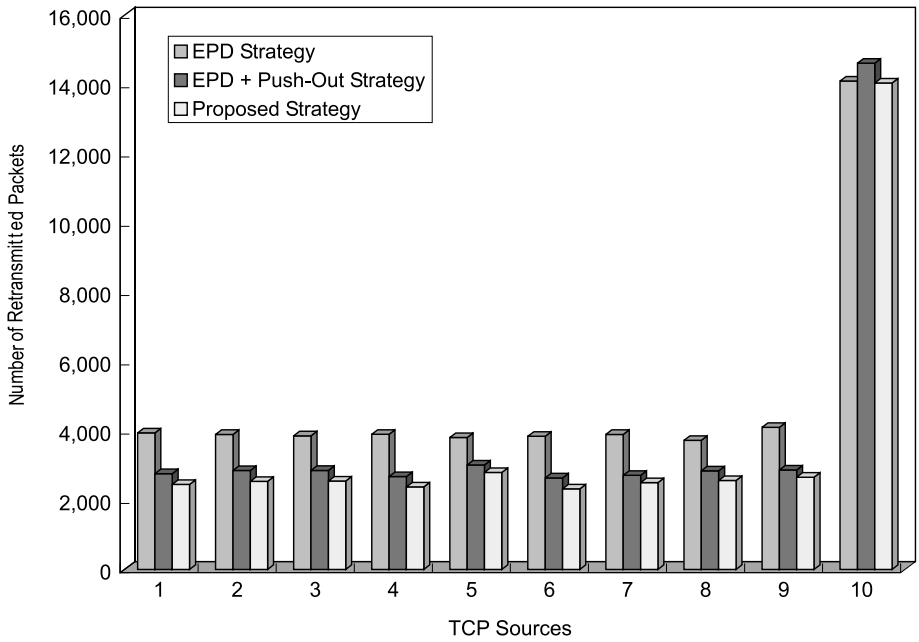


Fig. 10. The number of retransmitted packets with different sources.

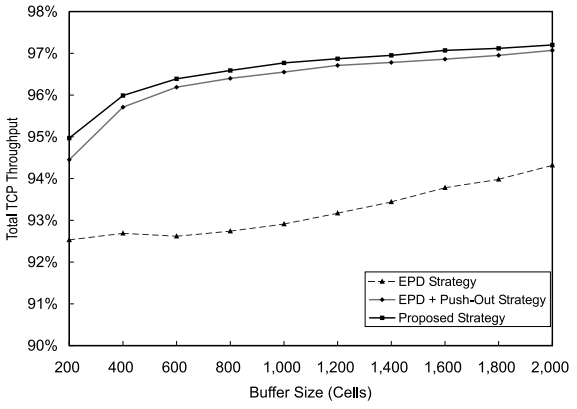


Fig. 11. The TCP throughput versus the buffer size.

Note that the total TCP throughput is not affected significantly by increasing the buffer size from 400 to 800 cells in the EPD strategy, since the sender will recover from N segment losses in N round trips with new Reno TCP. The increase of buffer size leads to a longer round-trip delay, which causes the counter effect that reduces TCP throughput. As the increase of buffer size is large enough, it can improve the TCP throughput again. Obviously, a larger buffer size reduces the number of dropped and corrupted packets, and tends to improve the TCP throughput.

4.2.5. Fair sharing

As discussed earlier, it is an important issue to provide fair sharing of available resources for GFR service. The fairness measure is based on the *fairness index*, which is defined as $(\sum x_i)^2/(n\sum x_i^2)$ [7], where x_i is the TCP throughput of the ith TCP source, and n is the number of TCP sources. If all sources share the available bandwidth more fairly, the fairness index will be much closer to 1. To further clarify the effect of the proposed scheme, we also enlarge the propagation delay between switches to 600 slot times (i.e., equivalent to about 500 km) and examine the performance.

Under FIFO discipline, the proposed strategy prevents the misbehaving source from obtaining an unlimited share of the bandwidth. The misbehaving source will be punished through the packet push-out buffering scheme. Our control approach almost achieves a rate-guaranteed service and fair sharing for TCP sources, as illustrated in Table 1. Since the growth of round-trip delay increases the cost of packet recovery, it in turn reduces the TCP throughput. However, with the proposed scheme, the ill-behaved source gets more significant punishment for longer round-trip delay.

4.3. Simulation for large network topology

We have examined TCP traffic with a single TCP per VC in previous subsection. However, in a real network the edge switch of the IP/ATM network may multiplex multiple TCP connections over a single VC. In this subsection, we assume that each VC carries multiple TCP connections and a more realistic scenario of GFR [19] is used for the simulation, as shown in Fig. 12. There are local IP/ATM edge switch pairs interconnected by two ATM backbone switches through GFR VC. Each GFR VC carries traffic of 10 TCP connections. The amount of allocated bandwidth for each VC is 10, 20, 30, 40 and 50 Mbps, respectively,

Table 1
The fair sharing of EPD, EPD+push-out and proposed strategies

TCP	1	2	3	4	5	6	7	8	9	10	Fairness index
EPD strategy											
Short	12.61	12.68	12.72	12.65	12.92	12.62	12.63	12.79	12.6	24.7	0.937
Long	11.73	11.81	11.58	12.19	11.85	11.96	12.10	11.88	11.87	16.17	0.989
EPD+push-out strategy											
Short	13.35	13.32	13.35	13.36	13.36	13.37	13.33	13.33	13.35	21.56	0.971
Long	12.71	12.61	12.63	12.76	12.74	12.82	12.42	12.78	12.59	13.74	0.999
Proposed strategy											
Short	13.49	13.43	13.45	13.48	13.44	13.5	13.41	13.43	13.42	21.42	0.973
Long	12.90	12.78	12.83	12.74	12.88	12.90	12.80	12.79	12.97	13.94	0.999

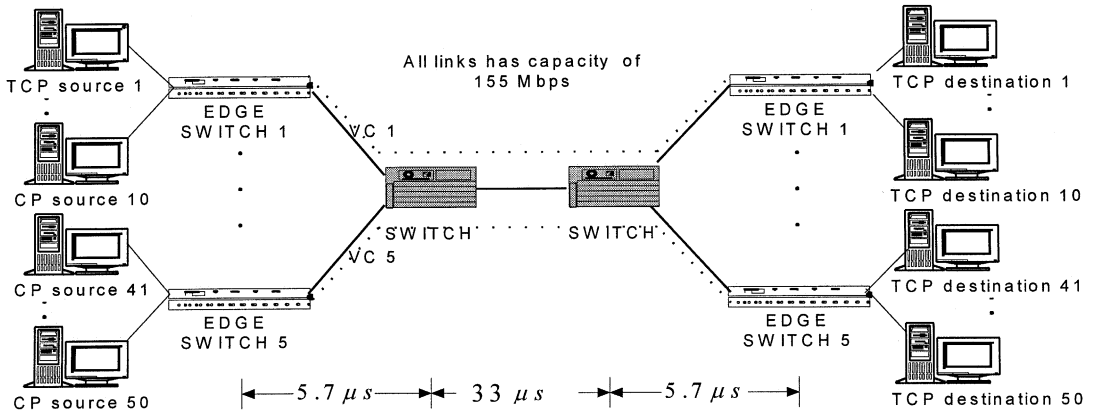


Fig. 12. TCP configuration of large topology (50 sources).

shared by the connected TCP sources. In addition, all sources are greedy TCP sources, i.e. they transmit data with twice of allowed transmission rate. The propagation delay is set to 1200 slot times (equivalent to approximately 1000 km). The remaining control parameters are the same to that in the small network configuration.

Since TCP segments from different sources are multiplexed on a single VC, the selective packet-discard strategy is unable to control the TCP rate accurately since the TCP source of the discarded packet in a VC may not be the malicious one. Thus it is difficult to achieve intra-VC fairness. In this scenario, the selective packet-discard strategy should be able to dominate the per-VC rate instead of per-TCP rate for inter-VC fairness.

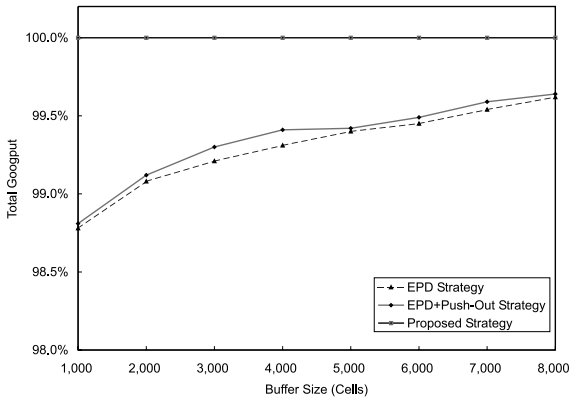


Fig. 13. The goodput versus the buffer size.

4.3.1. TCP goodput versus buffer size

The goodput performance is shown in Fig. 13. Again, the proposed scheme provides 100% goodput even with the multiplexed connections. Conversely, the multiple greedy TCP connections will makes the buffer occupancy high, and thus reduce the goodput of EPD-based strategies.

4.3.2. TCP throughput versus buffer size

As shown in Fig. 14, the throughput is improved significantly as buffer size increases. This is mainly because all sources are greedy. Additionally, push-out based schemes benefit more than the EPD strategy when the buffer space is sufficiently

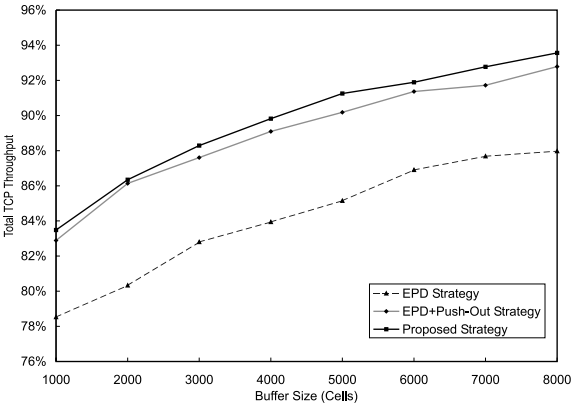


Fig. 14. The TCP throughput versus the buffer size.

large. In the EPD strategy, random drops may cause congestion to persist for a longer time; this results in further discarding and hence causes poor performance. Our proposed strategy provides greater buffer utilization than the EPD-based strategy by dynamically tracking the ABS, and thus offers better throughput.

4.3.3. Fair sharing

When a TCP source detects packet loss, the TCP CWND decreases, as does the transmission rate. Push-out based schemes drop packets from the worst behaved VC. However, the TCP source of the dropped packet may not be the most misbehaved one (i.e. the TCP source with the largest CWND/allowed_rate ratio). Since the packet discarding controls the rate less precisely, the fairness index of our scheme is around 0.98, which is little lower than in the small configuration simulation. However, since the packet dropping is random in the EPD strategy, the discarded packet may belong to another irrelevant VC, thus it causes the congestion to last longer. As a consequence, more packets not belonging to the worst behaved VC may be dropped; further unfair dropping may result, as shown in Fig. 15.

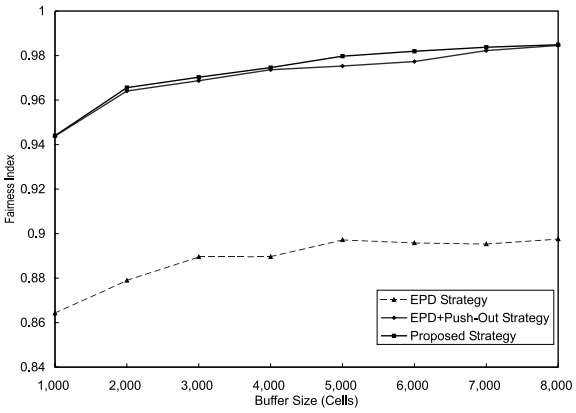


Fig. 15. The fairness index versus buffer size.

4.4. Simulation for chain network topology

In this subsection, we examine a chain topology, as shown in Fig. 16. The greedy UDP source is added into the topology and transmits data with twice of the allowed rate. The UDP and traffic of four TCP sources will pass through four switches and compete for resources with other TCP traffic. The propagation delay between two switches is 200 slot times (about 160 km), and the buffer size is 1000 cells. The throughput of both UDP source and TCP sources 1–9 are then analyzed.

4.4.1. Goodput, throughput and fair sharing

The total goodput is shown in Table 2. The total goodput of the EPD-based strategies are reduced. This is because the greedy UDP source keeps a high buffer utilization. It does not reduce the transmission rate since there is no flow control

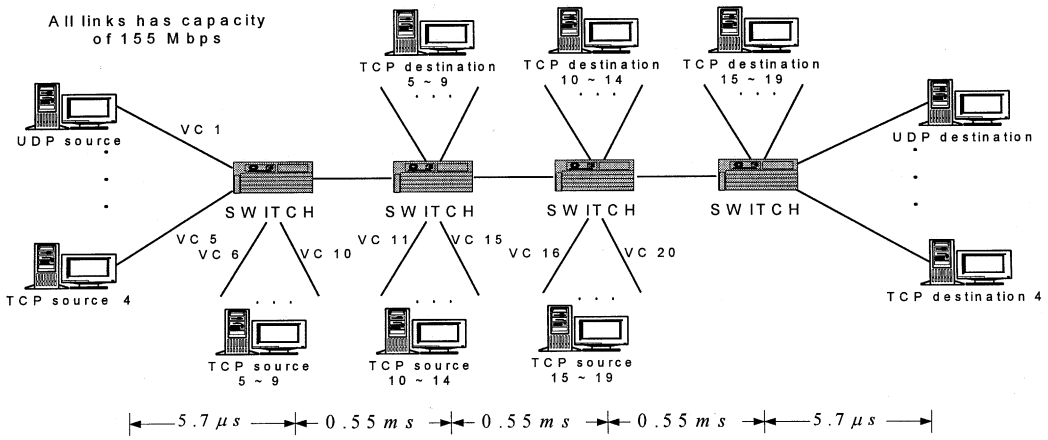


Fig. 16. TCP configuration of chain topology (20 sources).

Table 2
The goodput of EPD, EPD + push-out and proposed strategies

	EPD strategy	EPD + push-out strategy	Proposed strategy
Total goodput (%)	99.51	99.72	100

in UDP. Thus, it lowers the goodput of EPD-based strategies, which is similar to the phenomenon in the large network topology, as described in Section 4.3.

The per-connection throughput, the total throughput, and the fairness index are listed in Table 3. We can see that the UDP source gains much more bandwidth than other TCP sources, thus the throughput of TCP sources is greatly reduced. However, the proposed strategy can still provide relatively high throughput and fairness. The misbehaved UDP flow will be dropped by the push-out control scheme, and the throughput is improved by adopting dynamic threshold.

It is known that there is a bias for TCP flows with different round-trip time. In Table 3, the propagation delay of the TCP source traffic 1–4 is much longer than that of TCP sources 5–9. Again, the push-out control scheme minimizes the

Table 3
TCP throughput and fairness index of different control strategies

	UDP	TCP									Total throughput	Fairness index
		1	2	3	4	5	6	7	8	9		
EPD strategy	28.96	11.58	11.50	11.29	11.11	14.49	14.64	14.56	14.55	14.59	147.27	0.897
EPD + push-out strategy	26.91	12.00	12.13	12.18	12.20	14.47	14.34	14.54	14.47	14.45	147.69	0.925
Proposed strategy	26.44	12.39	12.41	12.48	12.34	14.37	14.30	14.30	14.42	14.33	147.78	0.932

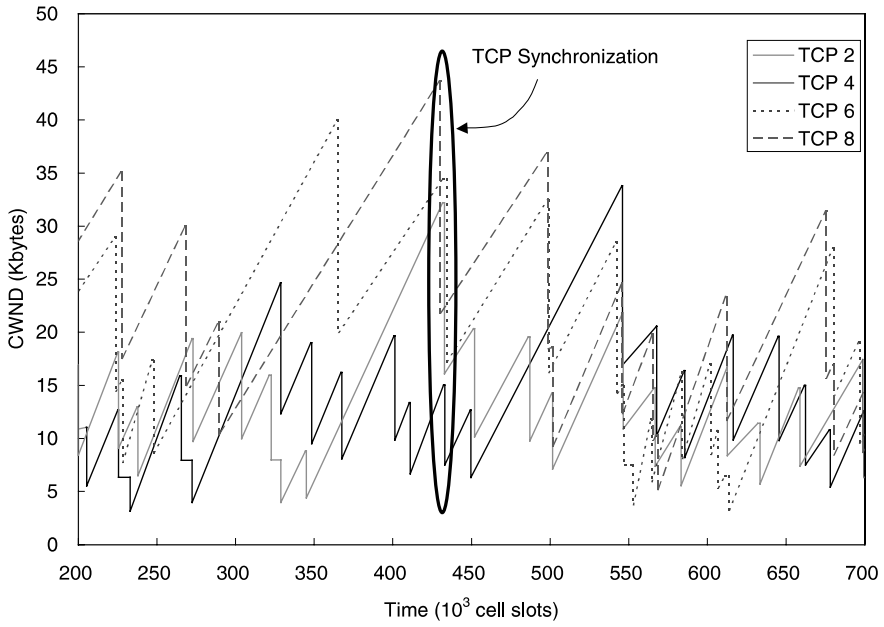


Fig. 17. The CWND of the EPD strategy.

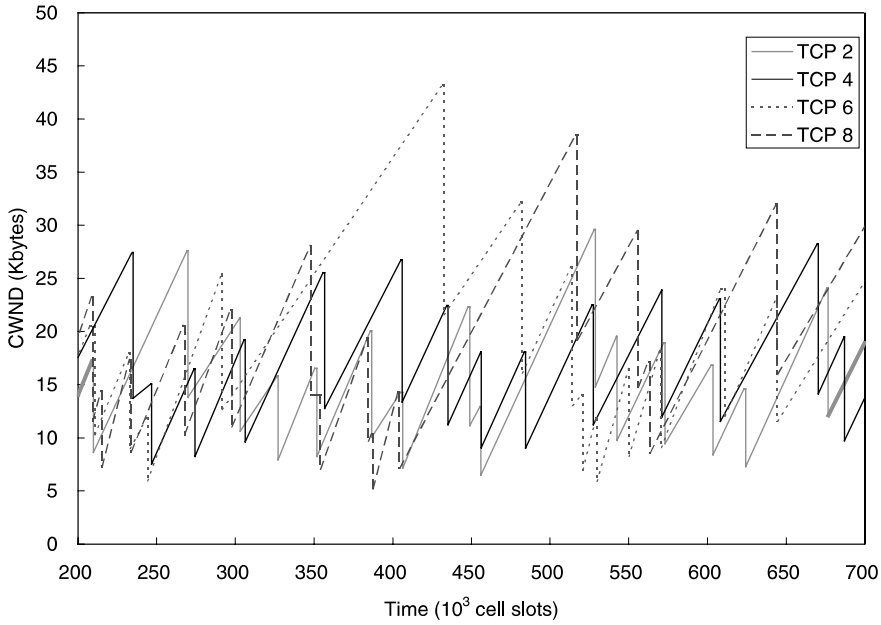


Fig. 18. The CWND of the proposed strategy.

difference of throughput for TCP flows with different round-trip times.

4.4.2. TCP synchronization problem

A good buffer management scheme should avoid the occurrence of TCP synchronization because it will degrade the throughput. The TCP synchronization might happen when the buffer overflow occurs. The packets of multiple TCP connections are dropped almost simultaneously and result in most TCP sources decreasing their CWND. As a result, the buffer utilization is significantly reduced. In the EPD strategy, there is no explicit scheme to control the misbehaved source; it just drops the incoming packets as buffer overflow occurs, and might result in TCP synchronization, as shown in Fig. 17.

In contrast, the proposed strategy always drops the packets of the most misbehaved source; hence the TCP synchronization can be avoided effectively, as shown in Fig. 18.

The simulation results show that our proposed approach achieves a minimum packet-level guarantee and fair resource allocation for GFR services in various network configurations. Moreover, it achieves 100% goodput and avoids the TCP synchronization problem. As a result, it brings the TCP throughput to a nearly optimal level.

5. Conclusions

The GFR services may be a key solution for supporting the TCP traffic over ATM networks. The implementation cost would affect the success of GFR service, however. The per-VC queuing is necessary to support the quality of GFR service, but may complicate the switch design. In this work, we have shown that it is possible to achieve the quality of GFR service through FIFO queuing. It is essential to have an efficient traffic control scheme to support the service requirements of GFR in network switches. Our proposed scheme consists of a selective packet-discard strategy and a packet push-out buffering scheme. This combination ensures the fair sharing of network resources and prevents a misbehaved connection from occupying excessive space in the FIFO queue. Although the push-out scheme complicates the buffer management, it achieves fair sharing of the

network resources and improves the total TCP throughput because it significantly reduces the number of packet retransmissions. To further improve the TCP throughput, a selective packet-discard strategy using ABS tracking is introduced. This strategy increases the buffer utilization that leads to a nearly optimal TCP throughput.

The simulation results show that our proposed approach achieves a minimum packet-level guarantee for packets and the fair allocation of resource for GFR services. Moreover, it achieves 100% goodput and brings the TCP throughput to a near optimal level. The proposed GFR traffic control approach can be implemented in most switch architectures with state-of-the art ASIC technology.

References

[1] N. Yin, S. Jagannath, End-to-end traffic management in IP/ATM internetworks, ATM Forum Cont. 96-1406, October 1996.

[2] R. Guerin, J. Heinanen, UBR + service category definition, ATM Forum Cont. 96-1598, December 1996.

[3] S.K. Pappu, D. Basak, TCP over GFR implementation with different service disciplines: a simulation study, ATM Forum Cont. 97-0310, May 1997.

[4] D. Basak, S.K. Pappu, GFR implementation alternatives with fair buffer allocation scheme, ATM Forum Cont. 97-0528, July 1997.

[5] R. Goyal, R. Jain, S. Fahmy, B. Vandalore, S. Kalyanaraman, GFR—proving rate guarantees with FIFO buffer to TCP traffic, ATM Forum Cont. 97-0831, September 1997.

[6] W. Stevens, TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms, Internet RFC 2001, January 1997.

[7] R. Goyal, R. Jain, S. Fahmy, B. Vandalore, X. Cai, Selective acknowledges and UBR + Drop policies to improve TCP/UBR performance over terrestrial and satellite networks, ATM Forum Cont. 97-0423, April 1997.

[8] A. Romanow, S. Floyd, Dynamics of TCP traffic over ATM networks, IEEE Journal on Selected Area in Communications 13 (4) (1995) 633–641.

[9] J.S. Turner, Maintaining high throughput during overload in ATM switches, Proceedings of the IEEE INFOCOM'96.

[10] M. Casoni, J.S. Turner, On the performance of early packet discard, IEEE Journal on Selected Area in Communications 15 (5) (1997) 892–902.

[11] P.-C. Wang, C.-T. Chan, Y.-C. Chen, An intelligent buffer management approach for GFR services in IP/ATM internetworks, IEEE ICON'99, Brisbane, Australia, September 1999, pp. 156–162.

[12] A. Demers, S. Keshav, S. Shenker, Analysis and simulation of a fair queueing algorithm, Journal of Internetworking Research and Experience (October 1990) 3–26.

[13] L. Zhang, Virtual clock: a new traffic control algorithm for packet switching networks, Proceedings of the ACM SIGCOMM'90, Philadelphia, PA, September 1990, pp. 19–29.

[14] A. Parekh, R. Gallager, A generalized processor sharing approach to flow control in integrated services networks: the single-node case, IEEE Transactions on Networking 1 (3) (1993) 344–357.

[15] A. Parekh, R. Gallager, A generalized processor sharing approach to flow control in integrated services networks: the multiple node case, IEEE Transactions on Networking 2 (2) (1994) 137–150.

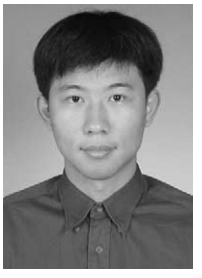
[16] S. Golestani, A self-clocked fair queuing scheme for broadband applications, IEEE INFOCOM'94, Toronto, CA, June 1994, pp. 636–646.

[17] H. Zhang, Service disciplines for guaranteed performance service in packet-switching networks, Proceedings of the IEEE 83 (10) (1995) 1374–1396.

[18] C.-T. Chan, Y.-C. Chen, S.-C. Hu, P.-C. Wang, A guaranteed fair queuing approach through FIFO buffering, IEEE Communication Letters 4 (6) (2000) 205–207.

[19] R. Goyal, R. Jain, S. Fahmy, B. Vandalore, Buffer management for the GFR service, ATM Forum Cont. 98-0405, July 1998.

Chia-Tai Chan received his Ph.D. degree in Computer Science and Information Engineering from National Chiao Tung University, Hsinchu, Taiwan in 1998. He is now with the Telecommunication Laboratories Chunghwa Telecom Co. Ltd. His research interests include the design, analysis and traffic engineering of broadband multiservice networks.



Pi-Chung Wang received his Ph.D. degree in Computer Science and Information Engineering from National Chiao Tung University, Hsinchu, Taiwan in 2001. He is now with the Telecommunication Laboratories Chunghwa Telecom Co. Ltd. His research interests include the Internet multimedia communications, traffic control on high-speed network and L3/L4 switching technology. He is a member of IEEE.

Yaw-Chung Chen received his Ph.D. degree in Computer Science from Northwestern University, Evanston, Illinois in 1987. During 1987–1990, he worked at AT&T Bell Laboratories as a member of Technical Staff. In August 1990, he joined the faculty of the Department of Computer Science and Information Engineering, College of Electrical Engineering and Computer Science, National Chiao Tung University, as an Associate Professor. He has been a Professor since 2000. His research interests include Internet traffic engineering, multimedia communications and high speed networking. He is a member of IEEE and ACM.