A Per-flow Based Node Architecture for Integrated Services Packet Networks

Dapeng Wu* Y. Thomas Hou† Zhi-Li Zhang‡ H. Jonathan Chao§ Takeo Hamada¶

Abstract

This paper presents a network node architecture and several traffic management mechanisms that are capable of achieving QoS provisioning for the guaranteed service (GS), the controlled-load (CL) service, and the best-effort (BE) service under the IETF integrated services (IntServ) paradigm. Our architecture offers the attractive feature of in-sequence delivery for all packets, albeit some of which may be out-of-profile. Simulation results show that, once admitted into the network, our architecture and traffic management algorithms provide hard performance guarantees to GS flows under all conditions, consistent (or soft) performance to CL flows under both light load and heavy load conditions, and minimal negative impact to in-profile GS, CL and BE traffic should there be any out-of-profile behavior from some flows.

1 Introduction

The IETF integrated services (IntServ) working group has specified three service classes, namely, the *guaranteed service* (GS) [2], the *controlled-load service* (CL) [5], in addition to the traditional *best-effort service* (BE). The GS guarantees that packets will arrive within the guaranteed delivery time, and will not be discarded due to buffer overflow, provided that the flow's traffic conforms to its specified traffic parameters [2]. The CL service is intended to support a broad class of applications which have been developed for use in today's Internet, but are sensitive to heavy load conditions [5]. The controlled-load service does not specify any target QoS parameters. Instead, acceptance of a request for controlled-load service implies a commitment by the network to provide the requester with a service closely approximating the QoS the same flow would receive under lightly loaded conditions.

To support the diverse QoS requirements from the GS, the CL, and the BE services in integrated services networks, new network architecture and traffic management algorithms must be in place. Such architecture and algorithms must meet the following performance criteria.

Criterion 1 (C1): For an admitted GS flow, the architecture and algorithms must ensure that the end-to-end delay bounds are never violated and packets are not lost if a source's traffic conforms to its traffic profile [2].

Criterion 2 (C2): For an admitted CL flow, the architecture and algorithms should provide, under all load conditions, a QoS closely similar to the QoS that the same flow would receive under lightly loaded network conditions [5].

Criterion 3 (C3): The network architecture and traffic management algorithms must be capable of controlling non-conforming GS/CL flows by minimizing their negative impact on other conforming GS/CL flows and BE flows [2, 5].

This paper presents a novel architecture and several traffic management algorithms based on per-flow queueing that satisfy the above three criteria to support integrated traffic of the GS, the CL, and the BE services. The main contribution of this paper is that it resolves the out-of-sequence problem associated with the non-conforming packets under an architecture proposed in [6] earlier. By employing the shaped virtual clock (SVC) scheduler and an marking/unmarking mechanism, the new architecture proposed in this paper is capable of achieving in-sequence packet delivery as well as meeting the three performance objectives even under the presence of non-conforming packets.

The remainder of this paper is organized as follows. Section 2 presents our per-flow based node architecture; the details of several traffic management algorithms are elaborated in Section 3. Section 4 uses simulation results to demonstrate the performance of our network architecture and traffic management algorithms. Section 5 concludes this paper.

2 Architecture

Figure 1 shows our architecture for the GS, the CL, and the BE traffic at each output port of a network node. Under our architecture (Fig. 1), we partition each output port buffer pool into three parts: one for GS flows, one for CL flows, and one for BE traffic. Within the same buffer partition for GS/CL/BE flows, we employ per-flow queueing for each individual flow.[1] Flows within the same buffer partition share the buffer pool of that partition while there is no buffer sharing across partitions.

Under the above buffering architecture, we design our per-flow based traffic management algorithms to achieve the three performance criteria.

*D. Wu is with Polytechnic University, Brooklyn, NY, USA.

†Y. T. Hou is with Fujitsu Laboratories of America, Sunnyvale, CA, USA.

‡Z.-L. Zhang is on the faculty of the Dept. of Computer Science, University of Minnesota, Minneapolis, MN, USA.

§H. J. Chao is on the faculty of the Dept. of Electrical Engineering, Polytechnic University, Brooklyn, NY, USA.

¶T. Hamada is with Fujitsu Laboratories of America, Sunnyvale, CA, USA.

[1] We employ per-flow queueing for BE traffic since it has been shown that TCP applications can achieve better performance under per-flow queueing than those under a common FIFO shared queue [4].

0-7803-6283-7/00/$10.00 © 2000 IEEE

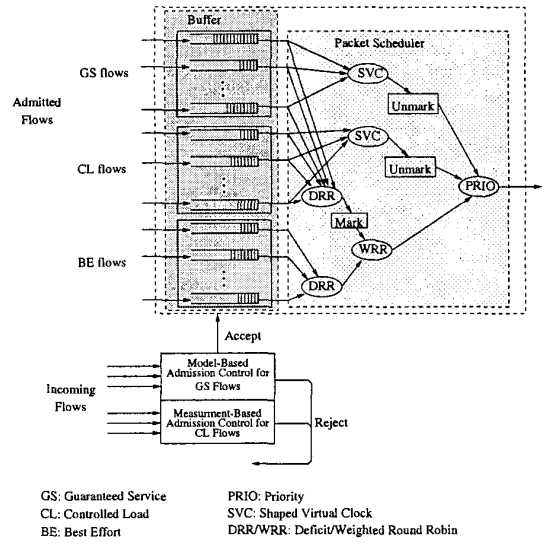

Figure 1: A per-flow based node architecture.

The first part of our architecture includes rate and buffer allocation, and packet scheduling. We follow the same approach in [6] to allocate rate and buffer for GS and CL services. To support the link sharing between the GS and the CL flows, we employ the Adaptive Rate allocation for Controlled-load (or ARC) [6] to allocate rate for GS and CL flows so as to provide hard bandwidth guarantee to GS flows under all conditions and consistent (or soft) bandwidth allocation to CL flows.

Also shown in Fig. 1 is a hierarchical packet scheduling architecture where a priority link scheduler is shared among a shaped virtual clock (SVC) for GS flows, a second SVC for CL flows, and a weighted round robin (WRR) for aggregate traffic from BE flows and out-of-profile GS/CL packets,[2] where BE flows and out-of-profile GS/CL packets are each serviced by a deficit round robin (DRR), respectively. Service priority is first given to the SVC scheduler for GS flows, and then to the SVC scheduler for CL flows. The WRR scheduler has the lowest priority in receiving service.

The GS/CL packets scheduled by SVC are all in-profile packets at this node. Thus, these GS/CL packets are unmarked (see Fig. 1). The GS/CL packets, which are scheduled by DRR, are out-of-profile at this node. Therefore, these GS/CL packets are marked (see Fig. 1). The reason why we use per-flow queueing and SVC scheduler for in-profile GS/CL packets is that it has been shown in [3] that SVC scheduling is able to provide bounds on the burstiness of session traffic (delay bounds) and enforce bandwidth allocation for each individual flow. The details of the SVC will be presented in Section 3.

The second part of our traffic management algorithms is call admission control (CAC), which we adapt the same approach in [6] and omit its discussion due to paper

[2]How to assign weight for WRR depends on the policy of network operators. If the network operators favor GS/CL users, they may assign a larger weight for out-of-profile GS/CL packets and a smaller weight for BE traffic, respectively.

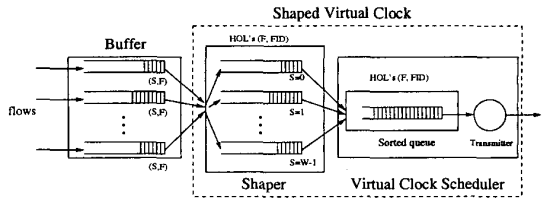

Figure 2: Structure of shaped virtual clock (SVC).

length constraint.

The last part of our architecture is on buffer management, or more specifically, packet discarding strategy when some buffer partition is full.

For GS buffer partition, since the CAC algorithm for an incoming GS flow includes buffer allocation, an admitted flow will have buffer space reservation throughout its path. Therefore, if all the GS packets are in profile, there should not be any buffer overflow for GS buffer partition. However, some upstream nodes may misbehave and transmit out-of-profile GS packets, which may potentially cause buffer overflow at downstream nodes. To address this problem, we propose a packet discarding mechanism called *selective pushout* (SP), which is presented in Section 3.2.

For CL flows, the buffer partition could also overflow since the traffic behavior of such flows is unpredictable and there is no reservation of buffer space for each CL flow. We propose a powerful pushout mechanism, called *selective pushout plus* (SP+) for packet discarding. In SP+, when the buffer cannot accommodate the incoming in-profile CL packet, out-of-profile packets will be pushed out first; if the free buffer space is still not enough after all out-of-profile packets are discarded, then in-profile packets from the quasi-longest queue will be pushed out. SP+ can control non-conforming flows as well as achieving fair buffer sharing among competing flows during congestion. The details of SP+ algorithm is given in Section 3.2.

For BE buffer partition, we use the dynamic threshold algorithm proposed in [1] to achieve fairness among BE flows.

3 Scheduling and Buffer Management

In this section, we elaborate the SVC scheduling algorithm and SP/SP+ packet discarding algorithms.

3.1 SVC for In-Profile GS/CL Packets

Figure 2 shows the conceptual structure of the buffer and the SVC scheduler. In the buffer, each flow has a logical queue. When a packet of flow i arrives, the packet is put into queue i in the buffer. Only the HOL packet of each queue will be stamped with a virtual start time S and a virtual finish time F. Based on the (S, F) pair, the SVC scheduler selects an HOL packet to be transmitted to the output link.

A SVC scheduler maintains a virtual system time $V(t)$, virtual start times $S_i(t)$ ($i \in \{1, \cdots, N\}$), and virtual finish times $F_i(t)$ ($i \in \{1, \cdots, N\}$), where N is the maximal number of flows that the system can support. $S_i(t)$ and $F_i(t)$ are updated upon arrival of the HOL

packet of flow i as follows:

$$S_i(t) = \max\{V(t), F_i(t^-)\}, \quad (1)$$

$$F_i(t) = S_i(t) + \frac{L_i^{HOL}}{r_i}, \quad (2)$$

where $F_i(t^-)$ is the finish time of queue i before the update, and L_i^{HOL} is the length of the HOL packet for queue i. The information $V(t)$, $S_i(t)$, and $F_i(t)$ are all represented with finite bits in implementation. Thus, without loss of generality, we suppose W is the maximum of $V(t)$, $S_i(t)$, and $F_i(t)$ that the system can represent.

The basic operations of the SVC are described as follows.

1. When a new HOL packet (its session index, say i) comes, the SVC computes the virtual start time $S_i(t)$ and virtual finish time $F_i(t)$ for this packet according to Eqs. (1) and (2). Then its (F, FID) pair is placed in the shaper queue, where FID denotes the flow identification of the HOL packet. Therefore, all the HOL packets are represented in the shaper queue.

2. The shaper maintains a logical queue for each discrete start time S. At every time slot, the shaper performs the eligibility test, and sends the (F, FID) pair(s) of those eligible packet(s), if any, to the scheduler queue. Specifically, if the current system time (modulo W) is equal to S, all the (F, FID) pairs in queue S, if any, will be moved to the virtual clock scheduler queue. In other words, only those that are eligible can be stored in the virtual clock scheduler queue.

3. The virtual clock scheduler prioritizes all eligible (F, FID) pairs based on their finish times F. It then chooses the packet with the smallest finish time to transmit first.

4. When an HOL packet (its session index, say i) is chosen to be transmitted and leaves the system, the scheduler queue removes its (F, FID) pair and selects another HOL packet, if any, to serve. In the meantime, if queue i in the buffer is not empty, this session can have another packet (regarded as a new HOL packet arrival) to join the shaper queue.

5. When an out-of-profile GS/CL packet is sent out by the DRR scheduler, its associated pair (F, FID) will be deleted in the shaper queue.[3] We assume that its associated pair (F, FID) can be found. In fact, we can use double linked list to implement the queue structure so that a pair (F, FID) can be easily found.

Note that packets do not pass the SVC but rather are kept in the buffer before being sent out to the output link. The packet selected by the SVC scheduler is transmitted directly to the output link without physically passing through the SVC.

[3]The (F, FID) pair of an out-of-profile GS/CL packet could not be placed in the scheduler queue since all packets in the scheduler queue are in-profile and will be scheduled at a higher priority than those served by the DRR scheduler.

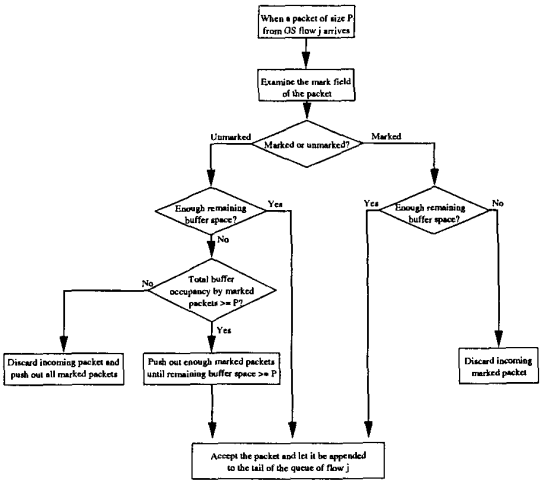

Figure 3: Flow chart of SP packet discarding mechanism for the GS partition.

3.2 Packet Discarding Mechanisms

An arriving packet is allowed to enter the particular buffer partition if there is enough remaining space for the buffer partition. Otherwise, we have to either discard the incoming packet or discard some other packet(s) in the buffer in order to make room for the incoming packet. In this section, we present two packet discarding mechanisms: one is called Selective Pushout (SP), which is employed for the GS partition; the other is called Selective Pushout Plus (SP+), which is employed for the CL partition.

Selective Pushout (SP) for GS Flows

Figure 3 shows the flow chart of the SP mechanism. According to Fig. 3, when an unmarked packet arrives at the node, SP makes every effort to let it enter the GS buffer. Specifically, if the remaining buffer space is not enough and total buffer occupancy by marked packets is larger than the size of the incoming packet, we randomly choose a queue with marked packets and push out enough marked packets from this queue; if all the marked packets in this queue have been discarded while the free space is still not enough, we randomly choose another queue which has marked packets and push them out, and so forth till the free buffer space can accommodate the incoming packet. On the other hand, when a marked packet arrives at the GS buffer, SP will let it join the buffer only if there is enough buffer space. Therefore, SP achieves the highest possible loss protection for in-profile (unmarked) GS packets.

In our implementation for SP, we maintain the following data structure for *each* GS flow (see Fig. 4). Each data unit in the GS buffer consists of a physical IP packet and three pointers, of which two pointers are used for doubly linked list L_{total} and the third is used for linked list L_{mark} as follows.

Linked list L_{total}: is a doubly linked list of all packets from a GS flow (both marked and unmarked). L_{total} is updated whenever an incoming packet is appended to the tail of the queue of the GS flow or

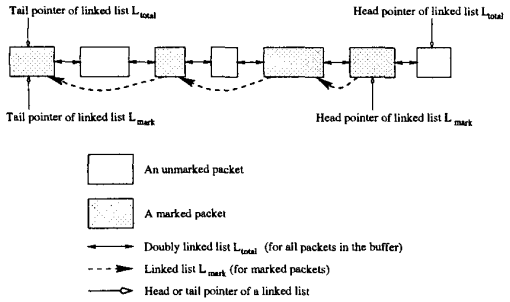

Figure 4: Linked list structure for GS/CL packets in the buffer.

a packet is served at the front of the queue of the GS flow by the output link.

Linked list L_{mark}: is the linked list of the marked packets *embedded* in the linked list L_{total}. L_{mark} is updated whenever an incoming marked packet is appended to the tail of the queue of the GS flow or a marked packet is either served by the output link *or* discarded by pushout mechanism.

By using the embedded linked list structure, out-of-profile GS packets can be easily found and discarded while the doubly linked list for L_{total} is able to preserve the connectivity of L_{total} when the packet at the head of L_{mark} is discarded. The introduction of the embedded linked list structure also solves out-of-sequence problem associated with the approach in [6], where out-of-profile GS packets are put into a separate queue and are serviced with a lower priority than those of in-profile GS packets.

Selective Pushout Plus (SP+) for CL Flows

Figure 5 shows the flow chart of the SP+ mechanism, where the queue length of flow i, $QL[i]$, is in unit of bytes. In our SP+ mechanism, a register is used to estimate the longest queue (LQ) in the CL buffer partition and is only updated upon the arrival and/or departure of a packet. How to update LQ upon the arrival of a packet is included in the flow chart of Fig. 5 (arrival updating). Similarly, when a packet from flow $i \in \mathcal{F}_{CL}$ departs from the output port, if $QL[LQ]$ is less than $QL[i]$, LQ will be updated to i (departure updating). The departure updating is not required but can increase the possibility of finding the actual longest queue.

Like SP mechanism, SP+ makes every effort to let the unmarked CL packet enter the CL buffer. The difference between SP+ and SP occurs when the remaining buffer space is not enough and total buffer occupancy by marked packets is less than the size of the incoming unmarked packet. In this case, if the incoming unmarked packet does not belong to the quasi-longest queue, we push out enough packets from the head of queue LQ till the free buffer space could accommodate the incoming packet. This mechanism is called Quasi-PushOut Plus (QPO+) [6]. The merit of QPO+ is that it tends to punish the user with (quasi) longest queue when the network is heavily congested.

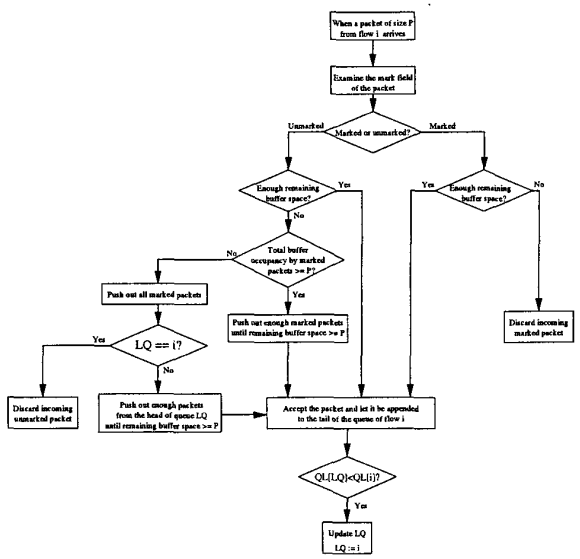

Figure 5: Flow chart of SP+ packet discarding mechanism for the CL partition.

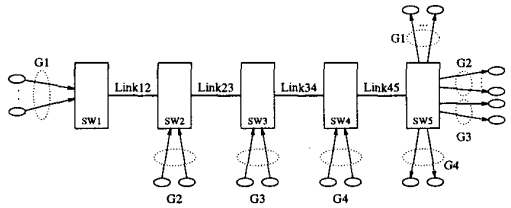

Figure 6: A parking lot network.

4 Simulation Results

This section presents simulation results to demonstrate the performance of our node architecture and traffic management algorithms.

4.1 Simulation Settings

We use the *parking lot* network configuration (Fig. 6) in our simulation study.

On the connection level, we assume that a GS or CL flow's inter-arrival times is exponentially distributed with an average of 50 seconds, with the holding time exponentially distributed with an average of 100 seconds.

The simulation parameters for the GS, the CL, and the BE services are shown in Table 1 [6]. For GS flows, we use the simple constant bit rate as their traffic pattern. This helps to simplify our simulations without any loss of generality in demonstrating the performance of our architecture and traffic management algorithms. For each BE flow, we use persistent TCP data traffic. For CL flows, we use an exponentially distributed on/off model with average $E(T_{on})$ and $E(T_{off})$ for on and off periods, respectively. During each on period, the packets are generated at its peak rate p_i, $i \in \mathcal{F}_{CL}$. The average bit rate for a CL flow $i \in \mathcal{F}_{CL}$ is, therefore, $p_i \cdot \frac{E(T_{on})}{E(T_{on})+E(T_{off})}$.

In Table 2, we list the simulation parameters at each

end system (i.e., sender and receiver) and network components (i.e., link and switch) [6].

Table 1: Simulation parameters for three types of services.

GS	Peak rate	1.5 Mbps
	Packet size	1K bits
	Delay bound	10 ms
CL	Peak rate	1.5 Mbps
	$E(T_{ON})$	2 ms
	$E(T_{OFF})$	2 ms
	Packet size	1K bits
	Packet loss ratio requirement	10^{-3}
	Delay bound	20 ms
BE (TCP)	Peak rate (light load)	1 Mbps
	Peak rate (heavy load)	10 Mbps
	Mean packet processing delay	300 μs
	Packet processing delay variation	10 μs
	Packet size	1K bits
	Maximum receiver window size	64K bytes
	Default timeout	500 ms
	Timer granularity	500 ms
	TCP version	Reno

Table 2: Simulation parameters at an end system and network components.

End system	GS	σ	15 packets
		ρ	1500 packets/s
		Buffer size	10 packets
	CL	σ	20 packets
		ρ	1000 packets/s
		Buffer size	10 packets
	TCP	Packet processing delay	500 μs
		Buffer size	500 packets
Switch	Buffer size	GS	500 packets
		CL	500 packets
		BE	1000 packets
	Packet processing delay		4 μs
	CL measurement window		100 Kbits
Link	Link speed		10 Mbps
	Distance	End system to switch	1 km
		Inter-switch	1 km

In our simulations, we set the target link utilization μ to be 0.90 in order to cushion any traffic fluctuation and measurement error.

We ran our simulator for 300 seconds simulation time and found that 50 simulated seconds are sufficient for our simulator to warm up. In order to obtain 95% confidence interval, we repeat each simulation eight times, each with a different seed.

4.2 Simulation Results

We organize our presentation as follows. Section 4.2.1 presents the performance of the GS, the CL, and the BE traffic under light and heavy load conditions and show that criteria C1 and C2 are satisfied. In Section 4.2.2, we show that our architecture and algorithms can effectively control non-conforming flows by minimizing their negative impact on other conforming flows (criterion C3). Section 4.2.3 compares our SP+ packet discarding with the tail-dropping mechanism.

Table 3: Number of GS, CL, and BE flows on each path under light and heavy load conditions in the parking lot network.

Path	Traffic Type	Number of Flows	
		Light Load	Heavy Load
G1	GS	1	1
	CL	1	2
	BE (TCP)	1	2
G2	GS	1	1
	CL	1	2
	BE (TCP)	1	2
G3	GS	1	1
	CL	1	2
	BE (TCP)	1	2
G4	GS	1	1
	CL	1	2
	BE (TCP)	1	2

4.2.1 Performance Under Light and Heavy Load Conditions

Figure 7 shows the link utilization at Link45 during the light and heavy load conditions. Table 3 shows the number of flows on each path under light and heavy load conditions in our simulation. The 95% confidence intervals for the maximum end-to-end delay (in ms) for GS and CL flows under light load are (1.586, 1.691) and (8.32, 9.28), respectively. The 95% confidence intervals for the maximum end-to-end delay (in ms) for GS and CL flows under heavy load are (1.718, 1.784) and (15.11, 15.83), respectively. We find that the delays experienced by each GS and CL flows are bounded and are less than the delay requirements for GS and CL flows, respectively. In Figs. 8 and 9, we plot the delay experienced by the GS flow and the CL flow traversing SW1 to SW5 (path G1) under light and heavy load, respectively. As shown in both figures, the delay experienced by this GS flow is bounded and is much less than its delay bound requirement (10 ms). For the CL flow, its delay is also bounded under both conditions and is less than its delay requirements (20 ms). As expected, there is some occasional delay increase for this CL flow under heavy load than under light load. Again, such increase is normal and is considered satisfying our performance objective for CL flows. Under both light and heavy load conditions, there is no packet loss from any GS or CL flow. In addition, we observe that there is no out-of-sequence phenomena for each flow.

Table 4: Performance of BE (TCP) traffic under light and heavy load conditions in the parking lot network.

BE Traffic (TCP flows)		Load Conditions	
		Light	Heavy
Throughput (kbps)	TCP1	33.2	11.2
	TCP2	32.8	10.9
	TCP3	32.5	10.5
	TCP4	31.7	10.1
	TCP5	–	11.2
	TCP6	–	10.9
	TCP7	–	10.5
	TCP8	–	10.1
Packet Loss Ratio (%)	Link12	0	0
	Link23	0	0
	Link34	0	2.5
	Link45	0	15.2

Table 4 shows the performance of BE flows under light and heavy load. We observe that the throughput of TCP1, TCP2, TCP3, and TCP4 all decrease under heavy load as expected. In contrast to GS and CL traffic, there is packet loss for BE traffic under heavy load conditions. We find such loss occurs at Link34 (output port of SW3) and Link45 (output port of SW4), respectively.

4.2.2 Control of Non-Conforming Flows

For those nodes that have policing mechanism, non-conforming GS/CL flows can be effectively controlled by marking out-of-profile GS/CL packets and discarding them when the corresponding buffer partition is full.

On the other hand, according to [5], network elements must not assume that each CL source or upstream elements have policing mechanism in place. Under such circumstances, the CL packets that are actually out-of-profile may not be marked at upstream elements/sources. We show that our architecture and algorithms can effectively control such non-conforming CL flows and thus achieve criterion C3.

We use the parking lot configuration under heavy traffic load for demonstration. The non-conforming flow is on path G4, which shares the bottleneck link (Link45) with all other flows on paths G1, G2, and G3. The non-conforming flow submits a peak rate of 1.5 Mbps as its traffic parameter for admission control but actually transmits at a peak rate of 10 Mbps. Since there is no policing mechanism for this flow at the entrance to the network, all out-of-profile packets from this flow are not marked.

Our simulations show that in the presence of such non-conforming CL flow, the contracted QoS to those conforming GS/CL flows can still be guaranteed while the non-conforming flow can be effectively isolated (due to per-flow queueing) and suffers from large packet loss rate (due to SP+ packet discarding). In particular, we plot the delay for a conforming GS and CL flows on path G1 in Fig. 10, which shows that the delays experienced by these conforming GS and CL flows are bounded and meet their respective delay requirements. Furthermore, the packet loss rate for these conforming flows remains zero during the simulation run. On the other hand, Fig. 11 shows that packets from the non-conforming CL flow suffer from heavy packet loss during the simulation.

The throughput of TCP connections (i.e., TCP1 through TCP8) are 8.2, 7.8, 7.6, 7.1, 8.2, 7.8, 7.6 and 7.1 kbps, respectively, which are comparable with those under heavy load in Table 4. So the non-conforming CL flow does not starve BE traffic.

4.2.3 SP+ vs. Tail-dropping

We compare the performance of SP+ with tail-dropping packet discarding scheme. Again, we use the same simulation settings in Section 4.2.2, except we discard the incoming packet when the buffer partition is full (tail-dropping) instead of SP+.

Figure 12 shows that under tail-dropping, even conforming CL flow experiences large packet loss. On the other hand, the same conforming CL flow experienced zero packet loss under SP+ packet discarding mechanism in Section 4.2.2.

5 Conclusion

This paper presented a per-flow based node architecture and traffic management algorithms, which have been demonstrated, to offer QoS provisioning for integrated traffic of the guaranteed service, the controlled-load, and the best-effort services under the integrated services paradigm. Our main contribution is that our architecture and traffic management algorithms not only meet the three criteria for integrated services networks, but also resolve the out-of-sequence problem for packet delivery, albeit the presence of non-conforming traffic flows. Simulation results showed that our node architecture and traffic management algorithms provide guaranteed performance for GS flows under all conditions, consistent (soft) performance for CL traffic under both light and heavy load conditions, and minimal negative impact on conforming flows when there are some non-conforming traffic flows.

References

[1] A. K. Choudhury and E. L. Hahne, "Dynamic queue length thresholds in a shared memory ATM switch," *Proc. IEEE INFOCOM'96,* pp. 679–687, March 1996.

[2] S. Shenker, C. Partridge and R. Guerin, "Specification of guaranteed quality of service," *RFC 2212,* Sept. 1997.

[3] D. Stiliadis and A. Varma, "A general methodology for designing efficient traffic scheduling and shaping algorithms," *Proc. IEEE INFOCOM'97,* April 1997.

[4] B. Suter, T. V. Lakshman, D. Stiliadis and A. K. Choudhury, "Design considerations for supporting TCP with per-flow queueing," *Proc. IEEE INFOCOM'98,* pp. 299–306, March 1998.

[5] J. Wroclawski, "Specification of the controlled-load network element service," *RFC 2211,* Sept. 1997.

[6] D. Wu, Y. T. Hou, Z.-L. Zhang, H. J. Chao, T. Hamada and T. Taniguchi, "On implementation architecture for achieving QoS provisioning in integrated services networks," *Proc. IEEE ICC'99*, pp. 461–468, June 1999.

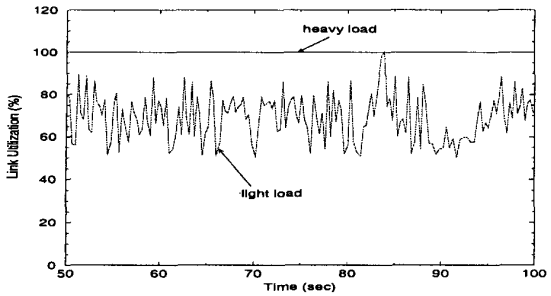

Figure 7: Link utilization of Link45 under light and heavy load in the parking lot network.

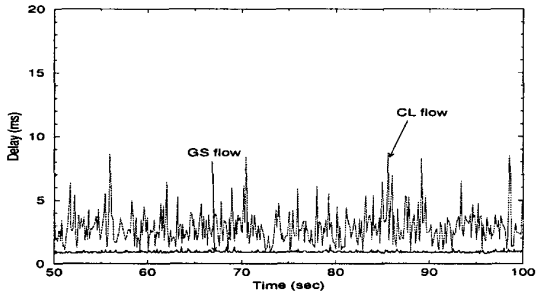

Figure 8: End-to-end delay of a GS flow and a CL flow under light load in the parking lot network.

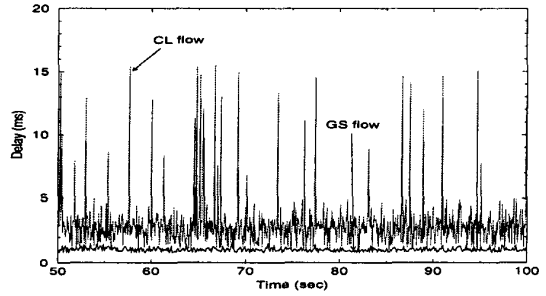

Figure 9: End-to-end delay of a GS flow and a CL flow under heavy load in the parking lot network.

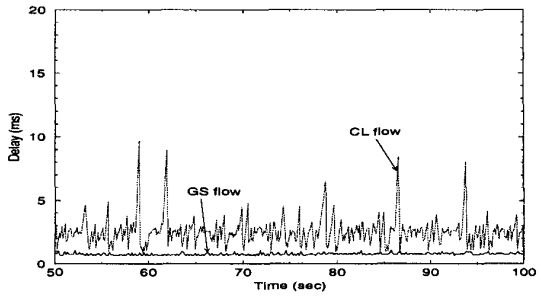

Figure 10: End-to-end delay for conforming GS and CL flows in parking lot network.

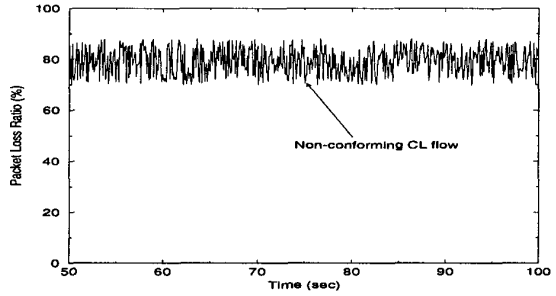

Figure 11: Packet loss ratio for the non-conforming CL flow in the parking lot network.

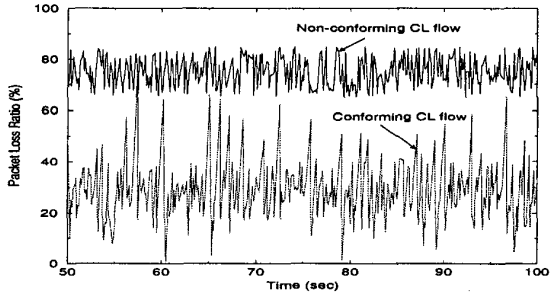

Figure 12: Packet loss ratio for conforming and non-conforming CL flows in the parking lot network under tail-dropping packet discarding mechanism.