Packet-Pair Rate Control — Buffer Requirements and Overload Performance

S. P. Morgan

S. Keshav

AT&T Bell Laboratories
600 Mountain Avenue
Murray Hill, New Jersey 07974-0636

ABSTRACT

Packet-pair is a feedback-based adaptive rate control scheme for high-speed wide-area data networks that use round-robin queueing at switches. Under packet-pair, each source continually estimates the effective speed of its bottleneck link by transmitting its data as pairs of back-to-back packets and measuring the separation of the returning pairs of acknowledgments. The source periodically adjusts its sending rate in an effort to avoid both overflow and underflow of its logical buffer at the bottleneck switch. The packet-pair scheme includes a timeout and retransmission strategy optimized to deal with losses.

This paper reports a simulation study of the good throughput achieved when a link is shared by a number of on-off sources, and the nominal offered load may be several times the actual capacity of the link. Some of the sources follow the packet-pair protocol while others do not. The simulations show no evidence of retransmission instability for packet-pair sources, at least up to nominal offered loads of 500%. Asymptotic goodputs in excess of 80% of the link capacity can be achieved with an aggregate switch buffer equal to one-half of a full-speed round-trip window. A switch buffer management strategy of dropping the entire longest queue when space is required gives packet-pair sources a substantial advantage over non-packet-pair sources in the presence of overload. Users therefore have considerable motivation to employ the packet-pair discipline, since they damage only themselves if they do not.

October 31, 1994

Packet-Pair Rate Control — Buffer Requirements and Overload Performance

S. P. Morgan

S. Keshav

AT&T Bell Laboratories
600 Mountain Avenue
Murray Hill, New Jersey 07974-0636

1. INTRODUCTION

Congestion control of data networks with high bandwidth-delay products is a much-studied problem. The bandwidth-delay product is a measure of the amount of data that can be put onto an end-to-end connection before a signal is received from either the network or the receiver indicating that something is amiss. The bandwidth-delay product is a lower bound on the amount of buffer space that must be provided at each queueing point in order that the full bandwidth may be used while also guaranteeing that the sudden onset of downstream congestion will never cause the buffer to overflow. If there are several independent users and we wish to guarantee that each user can instantly achieve a (user-specific) maximum bandwidth when other users are idle, while at the same time never experiencing any losses, each queueing point must be provided with buffer space equal to the sum of the bandwidth-delay products of all the users who can simultaneously share the queueing point.

In practice, networks are unlikely to attempt to guarantee users instant access to every scrap of available bandwidth, or to guarantee that there will never be any data lost owing to buffer overflow. Rather, the network designer will rely on statistical sharing to provide an acceptable quality of service (QoS) to users at the expense of a practicable amount of buffer space. The tradeoff between QoS and switch buffer size then becomes an important question.

Future integrated-service ATM networks are expected to carry several types of traffic with different QoS requirements. For example, constant-bit-rate, variable-bit-rate, and available-bit-rate (i.e., data) traffic can be handled on the same network using different priorities, However, to a first approximation, questions of congestion management, flow control, and buffer sizing can be considered separately for the various traffic types. For simplicity, in the present paper we shall consider only data traffic.

Data traffic is tolerant of delays but requires that, ultimately, every byte must be correctly received. Consider, for example, an output-buffered virtual-circuit switch with some fixed amount of buffer space at each output port and with individual sources subject to window flow control. If the instantaneous arrival rate at a given output port is greater than the output link speed, the switch may buffer arriving traffic until the buffer is filled and then discard overflowing cells, leaving the affected sources to time out and retransmit. Alternatively, the switch might accept buffer reservations for sources desiring to transmit bursts of data, so that a source holding a reservation would not be subject to overflow, while other sources seeking to transmit might be temporarily blocked.[1] A source would be expected to signal the end of a burst, and blocked sources would retry after some waiting period.

It is often convenient to measure the switch buffer space per output line in terms of the

[1] Burst reservations could in principle involve bandwidth rather than buffer space. However, on an integrated services network the instantaneous bandwidth available to data traffic may be more subject to preemption than the instantaneous buffer space.

bandwidth-delay product of the network (that is, the bandwidth of the network backbone times the edge-to-edge propagation delay across the network). In what follows, we shall sometimes call this product the *network round-trip window*. We would like to choose the size of the output line buffer so that the overflow probability, in the case of statistical buffer sharing, is some small number like 10^{-9} at a reasonable utilization level, or alternatively, in the case of burst reservations, the blocking probability is some small number like 10^{-6}. Simple statistical models indicate that if the source bandwidth, which is usually set by the source's access line speed, is comparable with the bandwidth of the network backbone, then the required buffer size may be as large as a few tens of network round-trip windows, while if the source bandwidth is small compared with the backbone bandwidth, the required buffer space may be as small as one network round-trip window or even less. This is in accord with the well-known fact that for a given amount of resource, greater statistical multiplexing gain is possible with low-speed users than with high-speed users.

In view of the possibly large buffer requirements of pure window flow control for high-speed data traffic, and the possible complexity of burst-level reservation protocols, there has been substantial interest in dynamic rate control schemes for wide-area networks. The idea of dynamic rate control is that each data source adjusts its sending rate periodically in response to congestion signals from within the network. A switch may generate a single bit of information indicating congestion [RAMA90] or several bits [MISH92]. Recent discussions in the ATM Forum have focused on the concept that the network will generate special resource management (RM) cells to inform sources of their currently permitted peak cell rate.

Most of the proposals for dynamic rate control have assumed FIFO queueing. If switches are able to implement round-robin service, we believe that the packet-pair protocol [KESH91, KESH94] is a particularly attractive rate-control scheme. In this protocol, data is typically transmitted in the form of packet-pair ''probes''; a singleton packet that cannot be paired with another packet on the same virtual circuit within a reasonable time is automatically sent by a timer. A packet-pair probe consists of two packets transmitted at access-line speed. Assuming round robin service at switches, the time separation between the two returning acknowledgments indicates the congestion at the bottleneck; more precisely, it indicates the transmission rate that the bottleneck node could sustain for the given virtual circuit at the instant the probe passed through it (alternatively, an RM cell could convey this information). After the initial probe returns, the source continues to transmit packet-pair probes, with the spacing between successive probes determined by the bottleneck rate that was reported by the latest returning probe. Ideally, at the bottleneck node the previous pair is departing just as the next pair arrives, and the maximum buffer requirement per virtual circuit is one packet-pair. In practice, the state of congestion of the bottleneck node will vary with time, and the objective, for a source with a backlog of data to transmit, is to keep the per-circuit buffer at the bottleneck node from either underflowing or overflowing. The source continually readjusts its transmission rate to drive the content of the bottleneck buffer toward a setpoint that is not too close to either end of the range.

Packet-pair relies entirely on measurements that can be made for each virtual circuit at the edge of the network. It has been analyzed and simulated elsewhere [KESH94] with respect to practical details such as start-up procedures, the optimal control of setpoints, and timeout and retransmission strategies. In the present paper we look particularly at the issues of stability under overload and the provision of firewalls.

Stability. It is well known that under heavy load, networks with finite buffers can exhibit retransmission instability. What needs to be demonstrated for a practical flow control scheme is that as the nominal offered load, defined as the aggregate load that all users would present if each user were on an otherwise idle network, increases without limit, the good throughput (''goodput''), defined as the total rate of transmission of useful information, becomes asymptotic to some reasonable fraction of the total network bandwidth.

Firewalls. The queueing discipline and the flow control scheme need to insure that the service given to well-behaved users is not degraded by the actions of careless or malicious users. We shall show, in fact, that if the network uses an appropriate switch buffer management

discipline. sources have an automatic incentive to adhere to the packet-pair rate control protocol, because they damage only themselves if they do not.

The rest of the paper is organized as follows. Section 2 shows the simulation scenario and the scaling. A set of independent sources go through on-off cycles, during which messages of exponentially distributed lengths are generated, lost packets are retransmitted, and exponentially distributed think times occur after transmissions are complete. The nominal offered load, defined as the sum of the utilizations that would be produced by the individual sources in the absence of any contention, is varied by manipulating parameters. Some sources use the packet-pair rate control protocol and others do not. Section 3 shows the results of extensive simulations using the REAL network simulator [KESH88]. Typical conclusions (Section 4) include the following. The simulations show no evidence of retransmission instability for packet-pair sources, at least up to nominal offered loads of 500%. Asymptotic goodputs in excess of 80% of the link capacity can be achieved with an aggregate switch buffer equal to one-half of a network round-trip window. Finally, an appropriate buffer management strategy at the switch gives packet-pair sources a substantial advantage over non-packet-pair sources in the presence of overload.

2. SIMULATION MODEL

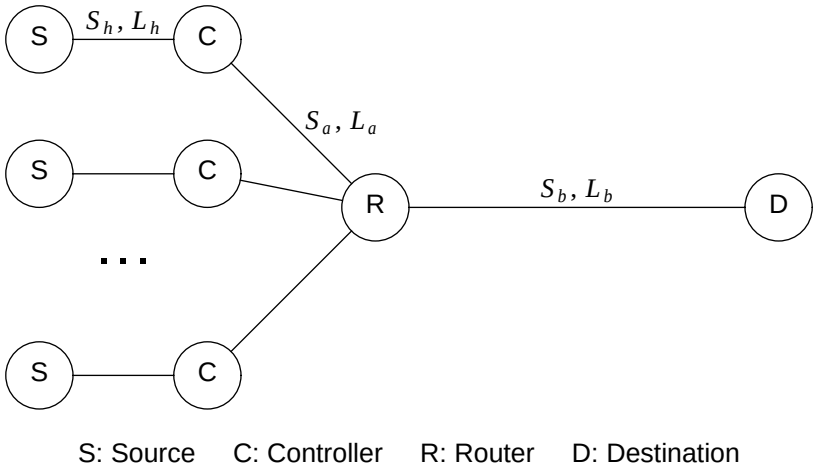

Figure 1. Simulation scenario.

The simulation scenario, shown in Fig. 1, is a closed queueing model with a number of statistically independent ON-OFF sources. A typical source S is ON for some random (but quantized) length of time. While on, it generates bits at a constant rate R_{ON}. The bit stream is divided into packets of equal length P and the packets are sent to a controller C over a link of speed S_h and latency L_h. (Latency = propagation delay.) For a packet-pair (pp) source, the controller imposes the packet-pair rate control discipline [KESH94] on the outgoing packets and puts them onto an access link having speed S_a and latency L_a. For a non-packet-pair (non-pp) source, the controller simply puts the packets onto the access link as they arrive.

Packets from all sources are multiplexed at a router R, which performs weighted fair queueing and puts the packets onto a backbone link of speed S_b and latency L_b. The router has a fixed total buffer size and maintains a separate logical queue for each source. It deals with overflow in one of two ways. Namely, when a packet arrives that would cause the total buffer space to overflow, the ‘‘drop last packet’’ discipline discards the arriving packet. Alternatively, the ‘‘drop longest queue’’ discipline discards the entire longest queue, whether or not that queue corresponds to the currently arriving packet. The ‘‘drop longest queue’’ discipline turns out to be effective in dealing with uncontrolled users.

The destination D returns an acknowledgment of length A for each packet that it receives. The controllers receive the acknowledgments and execute the same timeout and retransmission

strategy [KESH94] for both pp and non-pp sources, except that retransmitted packets from pp sources are subject to dynamic rate control, while retransmissions for non-pp sources are done as soon as the controller realizes that a retransmission will be necessary.

After a source has generated a message containing some number of packets, it goes OFF and waits until all the packets have been correctly acknowledged. It then waits for an additional, randomly chosen ''think time'', after which it again goes ON and repeats the cycle.

In the simulation model, ON times are exponentially distributed[2] with mean T_{ON}. OFF times are exponentially distributed with mean T_{OFF}. For simplicity we assume that the link between the router and the destination is the bottleneck, that is,

$$S_b \le S_a \le S_h \ , \tag{1}$$

and we assume that during an ON period, a source generates bits at a rate equal to the rate of the bottleneck link, that is,

$$R_{ON} = S_b \ . \tag{2}$$

Then when only one source is active, the round-trip time T_{RT}, measured from the instant that the head of a packet leaves the source until the instant that the tail of the acknowledgment reaches the source, is

$$T_{RT} = (P + A)\left[\frac{1}{S_h} + \frac{1}{S_a} + \frac{1}{S_b}\right] + 2(L_h + L_a + L_b) \ . \tag{3}$$

For the average length of an ON period, we take

$$T_{ON} = 2T_{RT} \ , \tag{4}$$

so that the average message length is

$$2R_{ON}T_{ON} = 2S_bT_{RT} \ , \tag{5}$$

that is, two round-trip windows.

The nominal offered load r_1 due to a single source is equal to the average utilization of the bottleneck link when only one source is present, i.e., when there is no contention. Thus, for the special case in which the ON rate of the source is equal to the speed of the bottleneck link,

$$r_1 = \frac{T_{ON}}{T_{ON} + T_{RT} + T_{OFF}} \ . \tag{6}$$

We can vary r_1 by varying T_{ON} and T_{OFF}, but it is clear that

$$0 \le r_1 \le \frac{T_{ON}}{T_{ON} + T_{RT}} < 1 \ . \tag{7}$$

The nominal offered load due to N statistically identical sources is Nr_1, and this can made as large as desired by suitable choice of N. The carried load cannot exceed 100% of the capacity of the bottleneck link, and the goodput is less than 100% if any packets have to be retransmitted.

For the present simulations, we chose:

$$P = 500 \text{ bytes} = 4000 \text{ bits} \ ,$$

$$A = 40 \text{ bytes} = 320 \text{ bits} \ ,$$

and for the line speeds,

$$S_h = 2\times10^9 \text{ b/s} \ , \qquad L_h = 0 \text{ s} \ ,$$

$$S_a = 8.0952\times10^7 \text{ b/s} \ , \qquad L_a = 0 \text{ s} \ ,$$

[2] Technically the distribution is geometric, because each ON period generates an integral number of packets.

$$S_b = 8.0952\times10^6 \text{ b/s}, \qquad L_b = 0.024706 \text{ s}.$$

From (3), the round-trip time is

$$T_{RT} = 0.050001\text{s},$$

and the number of packets in a round-trip window is

$$S_b T_{RT}/P = 101.19 \text{ packets}.$$

The timer for the transmission of a singleton packet is set at 0.1 s.

We chose the above numbers in order to have a reasonably large number of packets in a round-trip window and still achieve reasonable running times for the simulations,[3] since running times are approximately proportional to the number of cells the simulator has to handle. Translation from simulator numbers to real-world numbers is straightforward. Table 1 shows transcontinental round-trip windows at various backbone speeds, assuming a realistic round-trip propagation time of 50 ms. The round-trip window size in Mb and in ATM cells is shown in the second and third columns for backbone speeds ranging from 1.5 Mb/s to 622 Mb/s. The size of a simulated packet is approximately 1/100 of the round-trip window size.

Table 1

Round-trip windows at 50 ms

C_b Mb/s	W_b Mb	W_b cells
1.5	0.075	177
45.	2.25	5,310
155.	7.75	18,300
622.	31.1	73,300

We ran all the simulations with 10 statistically identical sources, that is,

$$N = 10.$$

Each simulation run was continued for a total of 420 simulated seconds, with intermediate results printed out at intervals of 20 simulated seconds. Since one round-trip time is 0.05 seconds, each print interval corresponds to 400 round-trip times. During each print interval, the bottleneck link could have carried about 40,000 packets at full utilization. We varied the nominal offered load of a single source from 0.05 to 0.5, so that the total nominal offered load on the bottleneck link varied from 0.50 to 5.00. The ''fair share'' of a single source varied from 0.05 to 0.10, inasmuch as under congestion no source would be expected to get more than 1/10 of the available bandwidth on average. Thus we would expect a typical source to have a goodput of somewhat less than 2000 packets to somewhat less than 4000 packets per print interval, depending on the nominal offered load. 2000 packets correspond to 10 average-length messages per source per print interval. We considered this adequate for statistical purposes.

3. SIMULATION RESULTS

The results of 168 simulation runs are plotted in Figs. 2-5. Each plot shows goodput vs. nominal offered load, but there are enough parameters so that some orientation is worthwhile.

Nominal offered load. In each plot, the nominal offered load varies from 0.5 to 5.0, that is, the

[3] Each point in Figs. 2-5 took approximately 45 minutes on a 150-MHz SGI Challenge M processor.

offered load per source varies from 0.05 to 0.5.

Mixtures of sources. We have considered four combinations, namely 10 pp sources, 9 pp and 1 non-pp, 1 pp and 9 non-pp, and 10 non-pp sources.

Goodput. Figures 2 and 3 show the total goodput for all four combinations of pp and non-pp sources, while Figs. 4 and 5 show the per-source goodput for sources of each type for the two cases in which pp and non-pp sources are mixed. *Buffer size.* We have considered three values of total buffer size, namely one round-trip window, 0.5 round-trip window, and 0.25 round-trip window. Each panel includes three curves, one for each value of the buffer size B expressed in terms of round-trip windows.

Overflow discipline. We have simulated both of the disciplines described in Section 2. In each figure, panels (a) and (b) represent drop-last-packet while panels (c) and (d) represent drop-longest-queue.

Detailed comparisons and comments on the results are given in Section 4.

We have not made a proper statistical analysis of the simulator output. We did go through the usual routine of taking a sample of n values of goodput corresponding to $n = 20$ successive simulation intervals, after discarding the first interval as a starting transient. We computed a sample mean $\overline{X}$ and variance $\hat{\sigma}^2$, according to [LAW82],

$$\overline{X} = (1/n)\sum_{i=1}^{n} X_i \ . \tag{8}$$

$$\hat{\sigma}^2 = \frac{1}{n(n-1)}\sum_{i=1}^{n}(X_i - \overline{X})^2 \ . \tag{9}$$

The intervals $(\overline{X} - 1.96\hat{\sigma},\ \overline{X} + 1.96\hat{\sigma})$ are plotted on Figs. 2-5. *If* the outputs of successive simulation intervals are uncorrelated,[4] and *if* the sample is large enough so that its mean is normally distributed, then the plotted intervals are ''95% confidence intervals''. We did not check any of this, therefore, the intervals shown on the figures should be regarded as only a rough indication of the scatter of the simulator output. It is not likely, however, that a more careful statistical analysis would upset the conclusions drawn in the next section.

4. CONCLUSIONS

The simulation results shown in Figs. 2-5 support several conclusions.

1. The curves of total goodput vs. nominal offered load in Figs. 2-3 show no sign of congestion instability, at least for offered loads up to 500%. Of course, a simulation study is not quite the same as a mathematical proof that the goodput has a nonzero asymptote as the load goes to infinity, but there is clearly not the catastrophic goodput collapse that occurs with unsophisticated retransmission schemes.
2. For a fixed offered load the goodput decreases, but not very fast, as the total buffer size decreases (see Figs. 2-3). This result helps to quantify the wide-spread belief that dynamic rate control should work satisfactorily with a total of less than one round-trip window of buffering. The buffer size needs to be at least $2N$ packets for N connections. In the present simulations, $2N$ packets would correspond to $B = 0.20$. Using smaller packets, packet-pair would undoubtedly function with still lower values of B. In an ATM network using AAL5, the spacing between cells of a single AAL frame could be measured at the receiver, reducing the buffer requirements still further.
3. For pure packet-pair traffic, drop-longest-queue (Fig. 2(c)) give somewhat lower goodput than drop-last-packet (Fig. 2(a)). This result is not surprising, inasmuch as drop-longest-queue involves dropping and retransmitting more packets than would have had to be retransmitted under drop-last-packet. It is perhaps surprising that the difference between

[4] Law and Kelton say [LAW82], *''It has been our experience that simulation output data are always correlated.''*

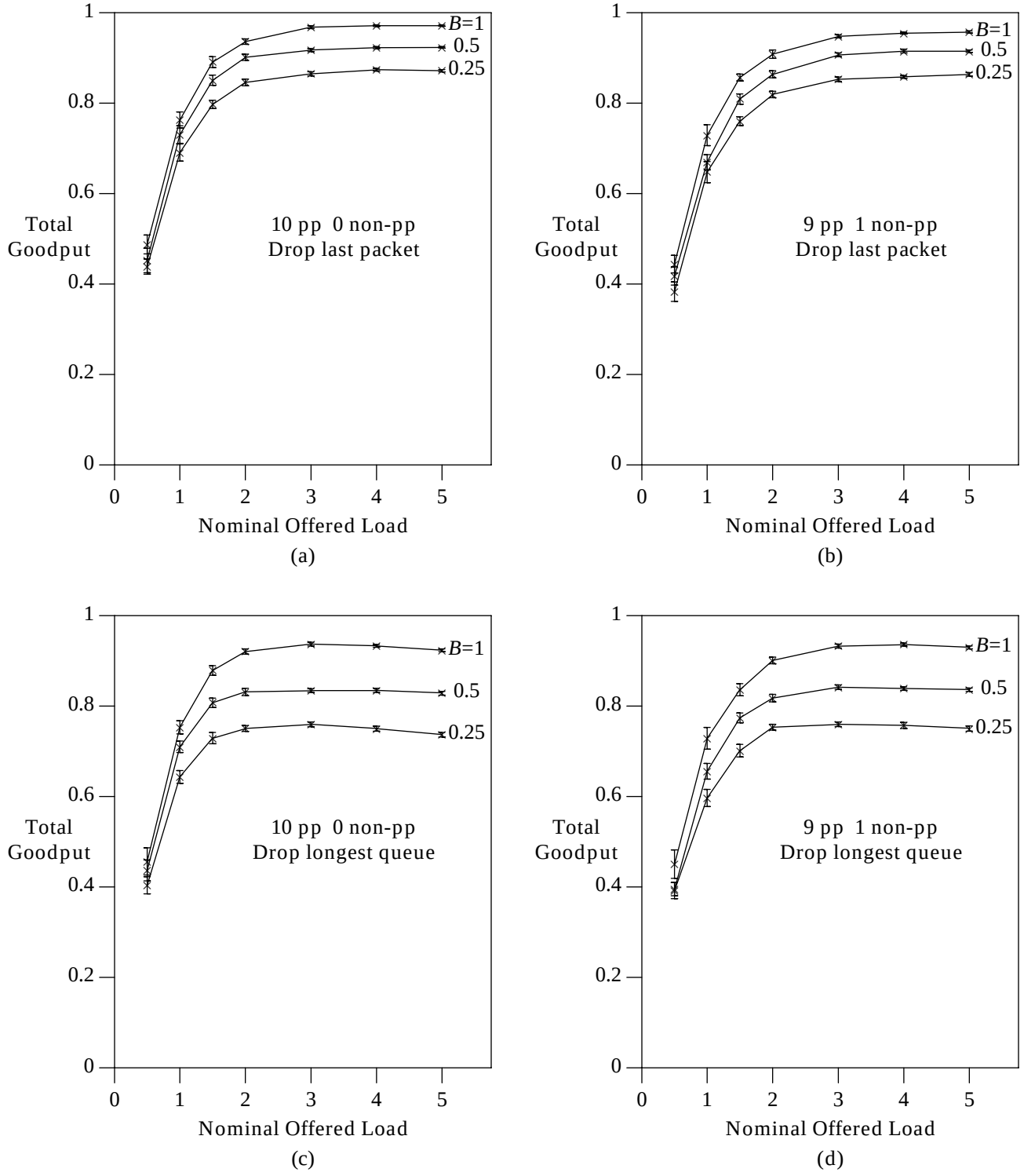

Figure 2. Total goodput vs. nominal offered load for pp plus non-pp sources.

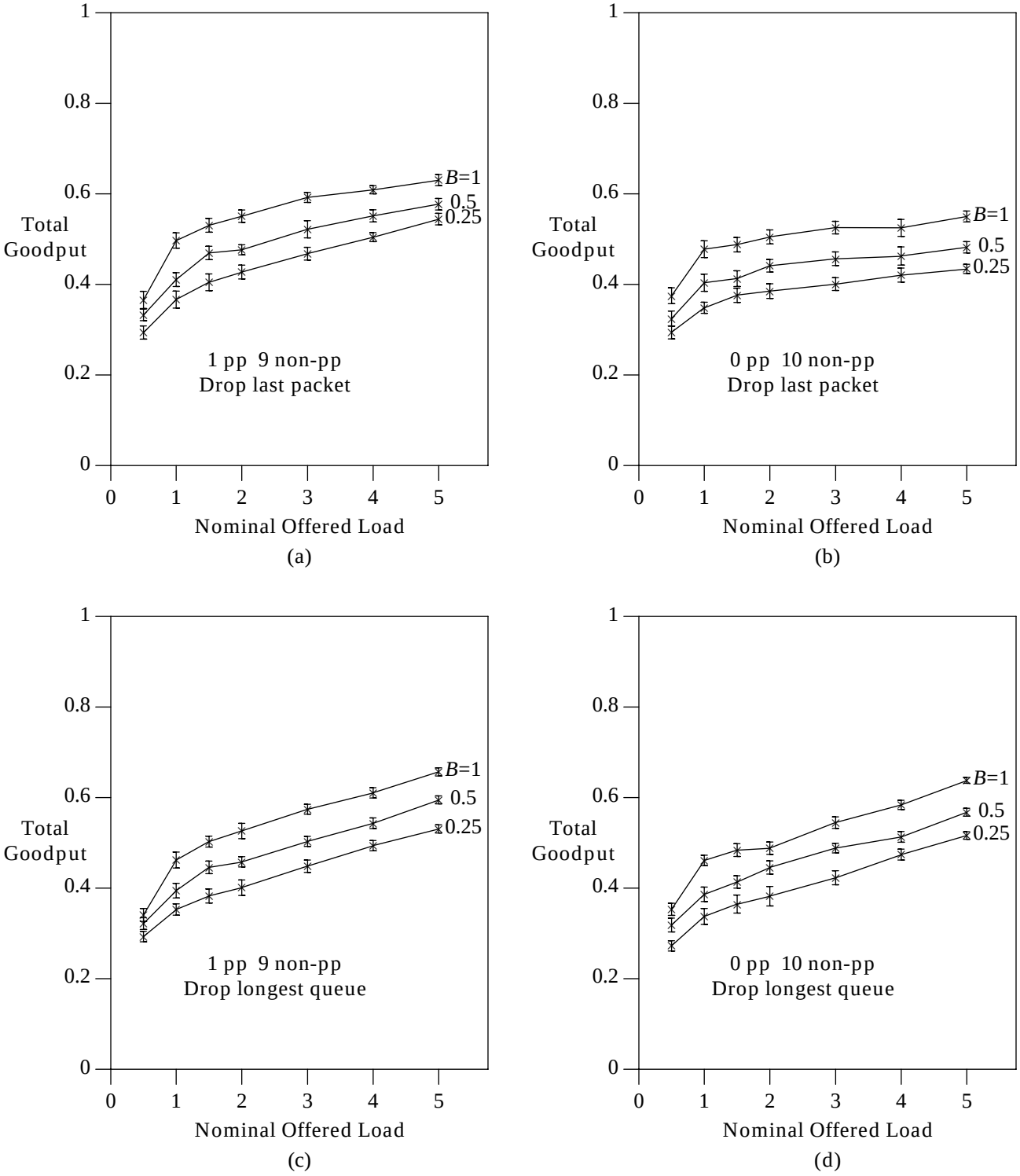

Figure 3. Total goodput vs. nominal offered load for pp plus non-pp sources.

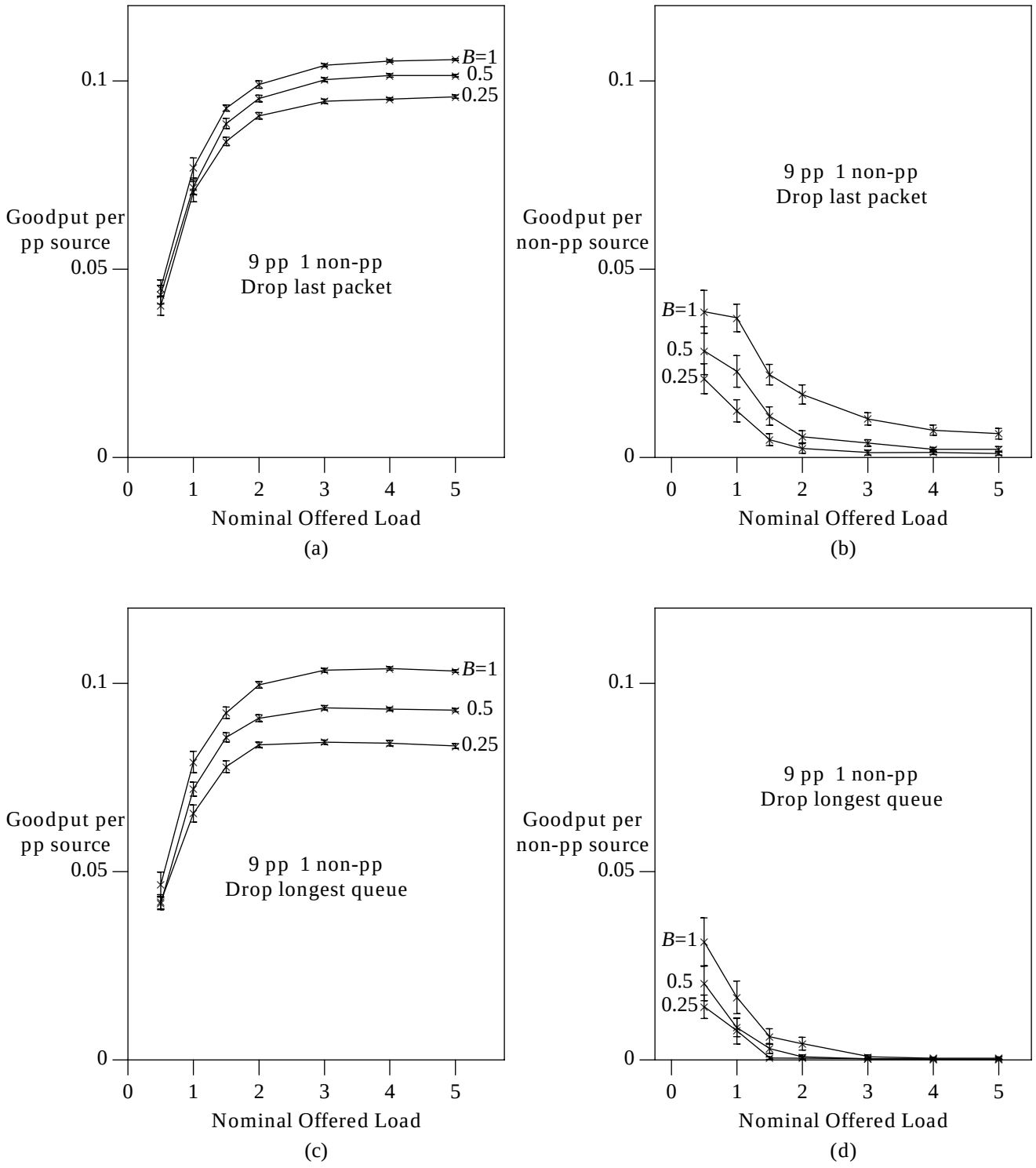

Figure 4. Per-source goodput vs. nominal offered load for pp and non-pp sources.

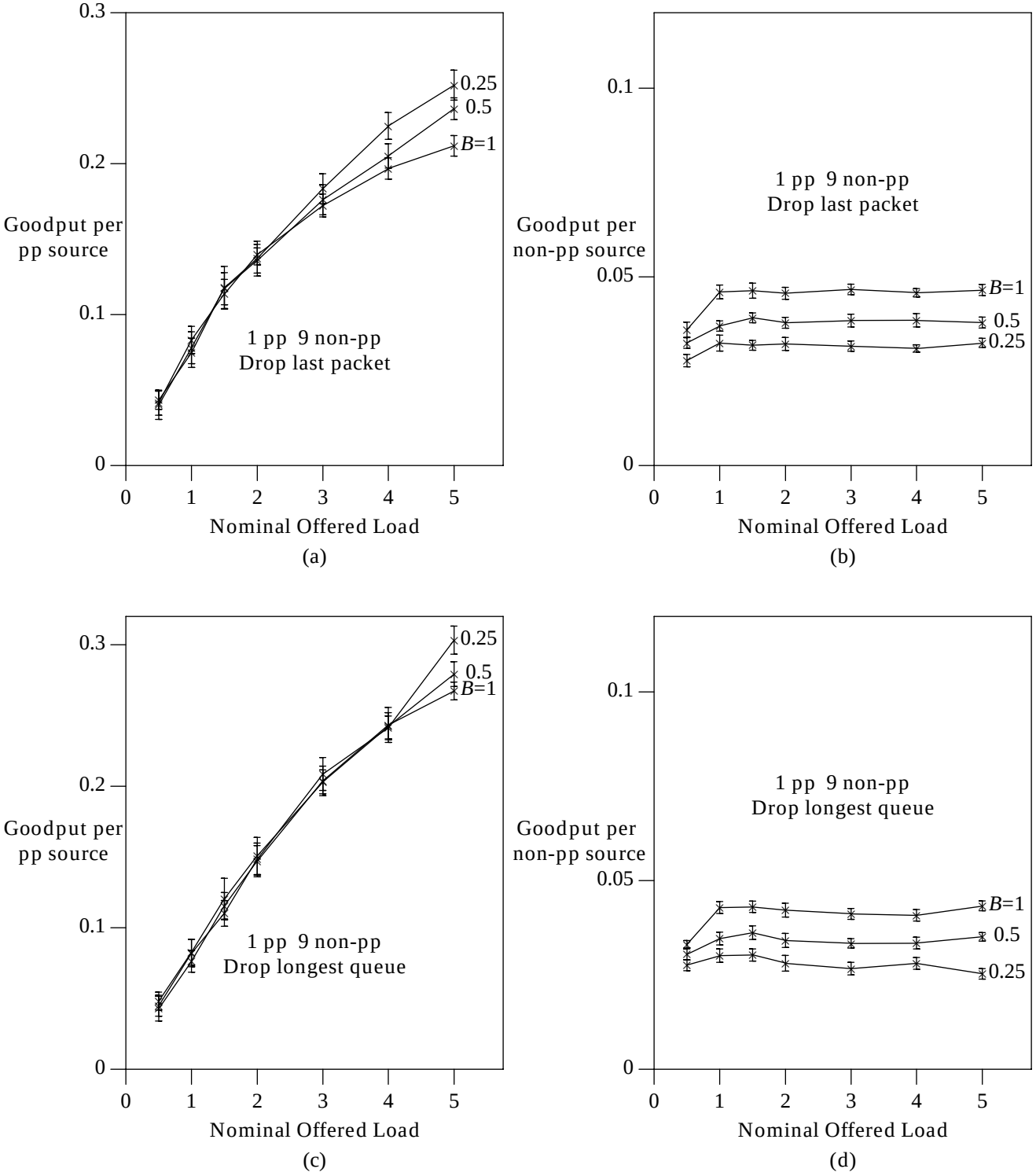

Figure 5. Per-source goodput vs. nominal offered load for pp and non-pp sources.

the two disciplines is no greater. As typical numbers, the asymptotic goodput for $B = 0.5$ is 92% for drop-last-packet and 83% for drop-longest-queue.

4. For pure non-packet-pair traffic (Figs. 3(b) and 3(d)), the goodput under heavy load gets to roughly 50%, which is not as good as pure packet-pair, but is by no means disastrous. The absence of congestion collapse for pure non-packet-pair traffic may be due to the fact that the non-packet-pair sources in our simulations use the packet-pair timeout and retransmission protocol [KESH94], which they can do because timeout and retransmission are independent of the algorithms used for rate control.

5. When packet-pair and non-packet-pair sources share the same bottleneck link, the packet-pair sources capture more bandwidth per source than the non-packet-pair sources, both when the packet-pair sources are in the majority (Fig. 4) and when they are in the minority (Fig. 5). Under heavy load, several packet-pair sources can essentially drive out a non-packet-pair source. This is particularly true under the drop-longest-queue discipline (Fig. 4(d)), where the non-packet-pair source keeps bumping into the buffer limit and losing its entire queue. When a single packet-pair source shares a link with many non-packet-pair sources, Figs. 5(a) and 5(c) show that the packet-pair source gets substantially more than its share, which would have been 10% of the congested link under the present scenario. Furthermore, the single packet-pair source actually does better with a smaller total buffer ($B = 0.25$), because it knows how to behave while the non-packet-pair sources do not.

The upshot of all this is that packet-pair is stable (for all engineering purposes) under congestion, that it can live with a fraction of a round-trip window of buffer space at each queueing point, and that the asymptotic goodput is in the 80-95% range, depending on buffer size. Perhaps most important, when packet-pair and non-packet-pair sources share a congested transmission path, the packet-pair sources have an advantage. This suggests that if a vendor advertises per-virtual-circuit queueing, round robin service, and a drop-longest-queue buffer management strategy, self-interest will lead users to use packet-pair rate control. If the network does not provide round-robin queueing but does give explicit rate feedback through the use of RM cells, a packet-pair source would simply use this rate instead of making its own rate estimates, and it would still use the timeout-retransmission strategy described in [KESH94].

REFERENCES

[RAMA90] K. K. Ramakrishnan and R. Jain, ‘‘A Binary Feedback Scheme for Congestion Avoidance in Computer Networks,’’ ACM Trans. Comp. Sys., Vol. 8, No. 2 (May 1990), pp. 158-181.

[MISH92] P. P. Mishra and H. Kanakia, ‘‘A Hop by Hop Rate-Based Congestion Control Scheme,’’ Computer Communication Review, Vol. 22, No. 4 (October 1992), pp. 112-123.

[KESH91] S. Keshav, ‘‘A Control-Theoretic Approach to Flow Control,’’ Computer Communication Review, Vol. 21, No. 4 (September 1991), pp. 3-15.

[KESH94] S. Keshav, ‘‘Packet-Pair Flow Control,’’ submitted to IEEE/ACM Transactions on Networking.

[KESH88] S. Keshav, ‘‘REAL: A Network Simulator,’’ Comp. Sci. Dept. Tech. Rep. 88/472, University of California, Berkeley (December 1988).

[LAW82] Averill M. Law and W. David Kelton, *Simulation Modeling and Analysis*, (McGraw-Hill, New York, 1982), pp. 145-151.