RC 20892 (06/24/97)
Computer Science/Mathematics 7 pages

IBM Research Report

Delay and Throughput Performance of Speeded-Up Input-Queueing Packet Switches

Roch Guérin

IBM Research Division
T. J. Watson Research Center
P. O. Box 704
Yorktown Heights, New York 10598

Kumar N. Sivarajan

Indian Institute of Science
Bangalore 560 012
INDIA

Research Division
Almaden · T.J. Watson · Tokyo · Zurich

Delay and Throughput Performance of Speeded-Up Input-Queueing Packet Switches

Roch A. Guérin and *Kumar N. Sivarajan*

Abstract—**In this note, we study the performance of high-speed packet switches that employ a combination of input queueing and output queueing. Specifically, we consider switches whose fabric runs at a speed greater than that of the input and output links. The purpose of this note is to highlight the result that a speedup factor of less than two for the switch fabric over the input link speed is sufficient to achieve full input link utilization (100% throughput) for many switch architectures. This result seems to have been overlooked by most researchers to date, who have often assumed that a speedup factor equal to the number of switch input ports n is necessary in order to achieve full link utilization. Consequently, we precisely specify the conditions under which a speedup factor of less than two is adequate. Furthermore, we also present simulation results on delay performance which show that a speedup factor of two is again sufficient to achieve nearly the same delay performance as that of an output queueing switch, even for bursty traffic. Finally, we consider the performance of switches with memory inside the switch fabric and see that such switches require even less speedup in order to achieve full link utilization.**

Keywords—**Input queueing, packet switch, speedup, throughput, ATM**

I. Introduction

Packet switching has become the predominant technology for high speed data networks and has even begun to be used for applications like voice which have traditionally relied on circuit switching. With the explosive growth that data networks have witnessed over the last few years, particularly the Internet, the design of high performance and low cost packet switches is of significant importance. A *packet switch,* consists of input and output ports connected to a switch fabric. The function of a packet switch is to transfer packets from the input ports to the appropriate output ports based on the addresses contained within the packets (in the packet headers). There are a few different options that are available for the design of the switch fabric. We refer the reader to the survey article by Ahmadi and Denzel [1] for a comprehensive discussion of these options. One is to use a *shared medium,* most commonly a bus, on which the packets from each input port are placed. Each output port selects the packets destined for itself from this shared medium by reading the packet headers. An example of a switch with such a fabric is PARIS [2], or its later version, plaNET [3]. A second option is a *crossbar* architecture in which one packet can be transferred to the output ports from *each input port simultaneously* provided the set of output ports are distinct. A third option is a *crossbar architecture with buffering* inside the switch fabric.

The key measure of the performance of a packet switch is its *aggregate throughput* which is the net rate at which it switches packets from its input ports to its output ports. In order for the switch to support the incoming traffic without dropping packets, the aggregate throughput of the switch must be at least equal to the sum of all the packet arrival rates on the input links. The aggregate throughput of the bus (shared medium) architecture is limited by the speed of the bus. In this architecture, the bus speed must be at least equal to the sum of all the input link speeds in order for no packets to be dropped. When input link speeds are already running into the hundreds of Mb/s, it is hard for the bus speed to satisfy this requirement. Thus the bus architecture is not suitable for high performance packet switches. In general, the difficulty of significantly speeding-up the switch fabric when links themselves are already running at very high speeds, is one of the main reason for the renewed interest in *input queueing* switches [4], [5], despite their well-known deficiencies [6], [7]. A goal of this paper is to emphasize that the speedup factor required to overcome those deficiencies is very moderate. In particular, in this paper we demonstrate this for crossbar or crossbar-like switch fabrics, with and without internal buffering.

In Section II we briefly review some results on the performance of input queueing switches and discuss some statistical models for the packet traffic. In Section III, we consider "input" queueing switches where the switch fabric runs faster than the input links. Here, we discuss the most significant result of this paper regarding the speedup factor needed to achieve full input link utilization. In Section IV, we consider the performance of crossbar fabrics with buffering inside the fabric and discuss how our results regarding speedup can be extended to them. In both Sections III and IV we also present simulation results regarding the queueing delay performance of these switches. Section V concludes the paper.

R. A. Guérin is with the IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598.

K. N. Sivarajan is with the Indian Institute of Science, Bangalore 560 012, India. This work was done while he was visiting the IBM T. J. Watson Research Center.

II. Input Queueing

We assume throughout this paper that the switch consists of n input and n output links (or ports), with all $2n$ links running at the same speed. We also assume that packets arrive independently on each input link and the statistics of the packet arrival process on all input links are identical. We further assume that the destinations of these packets are chosen uniformly randomly from among the n output links. The destination of a packet is independent of the destination of packets on other input links, but could possibly depend on the destination of packets on the same input link. Thus the n input links are statistically identical to each other and so are the n output links. We also assume that all packets are identical in size. This would be true, for example, in the case of links employing the Asynchronous Transfer Mode (ATM) where the term *cells* is used for packets.

For ease of description, we assume a slotted switch fabric that can simultaneously transfer packets from input ports to output ports in a *switch* time slot subject to the restriction that in each switch time slot, at most one packet *from* each input port and at most one packet *to* each output port can be transferred. A crossbar is a switch fabric that possesses this property. In other words, the set of packets transported through the switch fabric in a single time slot has distinct input ports and distinct output ports. If two input ports have a packet for the same output port, in the same switch slot, only one of them can be switched in that slot and the other waits in a queue at the input port. This waiting packet prevents any packets arriving later on that input link from being switched before it since packets arriving on an input link must be served in the order of their arrival. This phenomenon is traditionally referred to as *head-of-line blocking* and the packets at the heads of the queues at the switch inputs are called *head-of-line packets.* Let us denote the length of each switch slot by $1/R_s$ and the transmission time of a packet on any of the input links by $1/R_l$. Thus R_s and R_l are the transmission rates (in packets per second) of the switch and input links, respectively. Let us denote the expected number of packet arrivals on each input link per link slot by p. We term p the *input link utilization.* The expected number of packet arrivals on each input link per switch slot is then $q = \frac{R_l}{R_s}p$. If the input links and the switch fabric run at the same speed, $R_l = R_s$. This is the classic case of (pure) input queueing [6], [7] which we discuss in this section. When the switch fabric is speeded up, $R_l < R_s$, and the ratio R_s/R_l is termed the *speedup factor.* We treat this case in Section III.

We say that a switch input is *saturated* if it has a packet available for transmission through the switch fabric, in every switch slot. For example, when $q = 1$, the switch fabric is saturated (though the switch fabric can be saturated for $q < 1$). We define the *saturation throughput* of a switch fabric as the expected number of packets per input link that is transmitted by the switch in each switch slot, when all switch inputs are saturated. Clearly the saturation throughput will depend on the distribution of the packet destinations. As extreme cases, if all packets on all input links are destined to a single output port, the saturation throughput is $1/n$. When packets on input link i, $1 \leq i \leq n$, are always destined for output port i, the saturation throughput is 1. The saturation throughput has been widely studied in the case when, for each packet, each output port is equally likely to be the destination. As stated above, this is the case we assume in this paper. In this case, the saturation throughput can also be interpreted as the *output link utilization when the input links are saturated.* (Also recall our assumption of independent and statistically identical input links.)

Two cases have been studied in the literature. In the first case, the packet destinations of successive packets are independent. Both Hui and Arthurs [7] and Karol, Hluchyj and Morgan in their famous paper [6] showed that the saturation throughput in this case is $2 - \sqrt{2} \approx 0.586$, for large n. Thus, for large n, if the input links are operated at full speed, over 41% of the packets would have to be dropped, whatever be the size of the buffers used at the input ports. Clearly, the input links must be operated at less than full speed. In the second case, a burst of successive packets is destined to the same output port but the destinations of each burst are chosen uniformly from among the n output ports [8]. In this case, the burst sizes have the following *modified geometric distribution:* If the mean burst size is b, the size of a burst is $1 + B$ where B is a geometric random variable with mean $b - 1$. With this distribution, the first case is a special case of the second when $b = 1$. Li [8] has shown that the saturation throughput of the switch decreases monotonically with increasing burst size. He has derived an expression for the saturation throughput (for large n), which equals $2 - \sqrt{2} \approx 0.586$ for $b = 1$ and which tends to 0.5 for large b. We will use the notation, γ, for the saturation throughput; it will be implicit that γ depends on the distribution of packet destinations.

When the switch is not saturated we have to specify the process by which packets arrive on the links, in addition to the distribution of their destinations. We will assume that packets that cannot be transmitted through the switch immediately upon arrival, due to head-of-line blocking, are queued in an infinite buffer at each input link. We define the *stability throughput* as the maximum link utilization, p, for which these input queues are stable.

It has been implicitly assumed in much of the literature on input queueing that as long as the arrival rate of packets on each input link is less than the saturation throughput of the switch these input link queues are stable. For example, Iliadis and Denzel state in [9]:

"The system is stable as long as the average number of packets arriving in the input buffers is less than the saturation load. The process according to which packets arrive at the input queues has no effect on the stability of the system, although it does affect the delay characteristics."

However, that this is an assumption requiring proof has been recognized by Jacob and Kumar [10], [11] and their proof for the case of Bernoulli arrivals (successive packets have independent destinations) with identical rates on all the input links may be found in [12].

Saturation-Stability Property: For a specified distribution of the packet destinations, we will say that an arrival process satisfies the *Saturation-Stability Property* if the input link queues are stable whenever the expected number of arrivals on each input link per switch slot, q, is less than the saturation throughput, γ, of the switch. We will assume that the packet arrival processes arising in our discussion satisfy the Saturation-Stability Property.

The results from [6], [8] that we quoted above on the saturation throughput of an input queueing switch, together with the Saturation-Stability Property, imply that the input links (when running at the switch speed) must have an input link utilization, p, of no more than 50–60% depending on the burstiness of the traffic. Since link bandwidth is expensive (the demand has increased to match the speeds supplied by fiber-optics!) several methods have been devised to avoid idling such a large fraction of each link's bandwidth, and we review some of them in the next section.

III. Input Queueing with Speedup

One option is to run the switch fabric n times faster than each input link so that it is capable of transferring every arriving packet to its output port, regardless of the distribution of the destination ports of the packets, i.e., there is no head-of-line blocking. Note that in this case, buffers must still be provided for packets awaiting transmission if the output link speed is lower than that of the switch fabric. (Typically the output link speed is equal to the input link speed, as assumed in this paper, though this need not always be the case.) This is the classic case of output queueing which is discussed in [6]. In particular, since there are no input queues, output queueing achieves a (stability) throughput of 100% so that the input links can be operated at full utilization without any packets being dropped (if sufficient buffers are provided at the output ports).

However, as mentioned earlier, when the link speeds are already high, running the switch fabric n times faster becomes nearly infeasible. This is the same problem that is faced by the bus architecture (Section I). Thus, considerable effort has been directed towards crossbar architectures whose performance approaches that of output queueing, while requiring the smallest possible speedup factor. One of the early efforts in this direction was the Knockout Switch [13] proposed by Yeh *et al*. In this switch, in each time slot, upto $L > 1$ packets can be transferred to each output, instead of one packet as in a pure input queueing switch. This switch can be considered to have a *parallelism factor* $L > 1$. If the parallelism factor $L = n$, the number of input ports, then this switch is identical to an output queueing switch, and if $L = 1$, it is identical to an input queueing switch. The performance of switches with an arbitrary parallelism factor, L, where $1 \leq L \leq n$, has been analyzed in [13], [14], [15]. In particular both Oie *et al.* [14] and Chen and Stern [15] have shown that the throughput of the switch increases rapidly with L and is greater than 99.5% per input port for $L = 4$, even when n is arbitrarily large.

The results regarding the throughput of switch fabrics like the Knockout Switch with a parallelism factor $L > 1$ have given risen to the "folk theorem" that a *speedup* factor of n is *necessary* in order to achieve a throughput of 100%. However, while this *may be true* for certain switch fabrics that obtain the speedup by employing parallelism in space, as in the case of the Knockout Switch, it does not apply to a crossbar switch fabric running at a faster clock speed than the input links, as we will now see.

We remove the assumption that $R_l = R_s$ made in the previous section and ask: *For a given input link speed R_l what is the switch speed R_s required in order to achieve an input link utilization of unity, i.e., 100% (stability) throughput?* Note that when $R_s > R_l$, queues must be provided for at the output links too for packets to await transmission. Due to our assumptions, specifically the symmetry among the output links and the equality of the transmission speeds on the input and output links, the stability of these output queues is assured. Our answer is embodied in the following proposition.

Proposition 1: The input link utilization of an input queueing switch fabric running at a rate $R_s > R_l/\gamma$, where R_l is the rate on the input links and γ is the saturation throughput of the switch with the same distribution for the packet destinations as that of the packets arriving on any input link, can be made arbitrarily close to one, provided the arrival process satisfies the Saturation-Stability Property.

Proof: The expected number of arrivals on each input link *per switch slot* is $q = \frac{R_l}{R_s}p$. If $R_s > R_l/\gamma$,

$q < \gamma p < \gamma$ for all p, and the input queues are stable since the arrival process satisfies the Saturation-Stability Property by assumption. Thus the input link utilization p can be made arbitrarily close to 1 and the switch achieves a (stability) throughput of 100%. ■

The reason why a parallelism factor $L = 2$ is not the same as a switch speedup factor of two, in time, is as follows. For a switch with a parallelism of two, all the packets that are switched in a single slot *must be from different input links.* When the speedup is in time and the switch runs at a clock speed faster than that of the links, we have two kinds of slots: link slots which are equal to the transmission time of a packet on an input link, and switch slots which are equal to the transmission time of a packet through the switch fabric. When the switch runs at twice the clock speed, there are two switch slots for every link slot. In this case, two packets from the *same input link* can be switched by the fabric in these two switch slots. This cannot be done with a switch fabric running at the link speed but with a parallelism of two. This seemingly innocuous difference in the two models thus has a very significant consequence.

However, note that full input link utilization can also be achieved by using a parallel switch fabric, provided the operation of one of the fabrics is *staggered* from that of the other by half a slot. This result is alluded to by Hui and Arthurs [7, page 1268].

"... we may have more than two switch planes operating in parallel, provided adjacent switch planes are staggered by a period of time ... Effectively, the throughput capacity of the switch with S planes is equal to S times the single plane switch. ... increasing the throughput of the switch increases throughput on the output trunk beyond the 58% capacity for random traffic through 1 switch plane to full loading ..."

However, probably because this result was not the focus of this paper, this point seems to have been overlooked by many of the researchers following them. For example, Oie *et al.* [14], Chen and Stern [15], and Pattavina and Bruzzi [16] have all undertaken fairly detailed analyses of switches with small parallelism factors, ignoring the alternate approaches to switch speedup such as using staggered switching planes as proposed in [7] or running the switch fabric at a clock speed higher than that of the input links. However, Liew [17] in his extensive simulation study of packet loss probabilities under various switch architectures, has observed: *"We think there is no reason in the speedup scheme to intentionally limit the number of clearable packets from each input port to one."* From his simulation results he observes that for a speedup factor of 2: *"... neither speedup scheme has maximum throughput limitations ...."* But since his focus was on loss probabilities and several different input and output queueing schemes were considered, the consequences of this observation were not highlighted or explored.

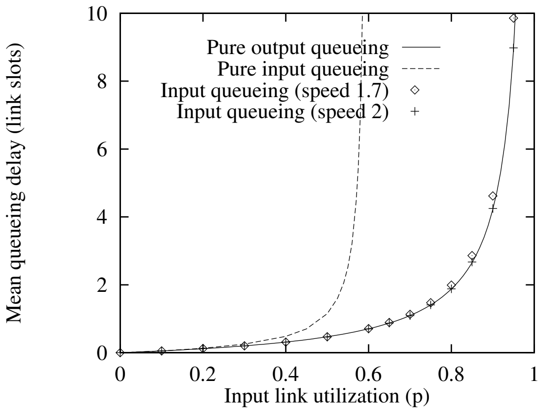

Fig. 1. The delay performance of an input queueing switch with speedup factors of 1.7 and 2 obtained by simulation is compared with the delay performance of pure input queueing and pure output queueing. The mean burst size $b = 1$ so that the destinations of successive packets are independent. The expected number of packets in each link slot equals p.

We illustrate this result in Figure 1, where we have plotted the mean queueing delay experienced by an arbitrary packet versus the expected number of packets per link slot, p. The queueing delay includes not only the delay in the input link queue but also the delay in the output queue. (Recall that the output links run at the same speed, R_l, as the input links.) In this figure, we have assumed that $n = 16$ and the mean burst size $b = 1$ so that the destinations of successive packets are independent. The saturation throughput in this case is 0.6 (the result of [6] of 0.586 is for large n and the saturation throughput for finite n is larger) so that a speedup of $1.7 > 1/0.6$ should be sufficient to achieve full input link utilization, from Proposition 1. We can see that this is indeed the case. However, note that while this speedup factor is sufficient to achieve the same full input link utilization as output queueing, we expect, *a priori,* the delay performance to be worse than that of output queueing. While this is true, *the surprising result is that the queueing delay even at this modest speedup factor is close to that of output-queueing,* as can be seen from Figure 1. When the speedup factor is increased slightly to 2, the mean queueing delay performance is almost indistinguishable from that of output queueing.

To gain some intuition into this unexpected result, we separately plot the mean queueing delays in the input and output queues for both speedup factors (1.7 and 2) in Figure 2. We see that the queueing delays in the output queues dominates the total queueing delay; the queueing delay in the input queues is negligible. Thus, even a modest speedup factor quickly overcomes the head-of-

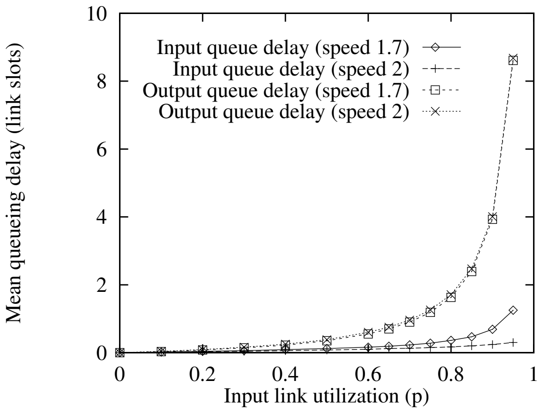

Fig. 2. The mean queueing delays in the input and output queues for the two speedup factors considered in Figure 1.

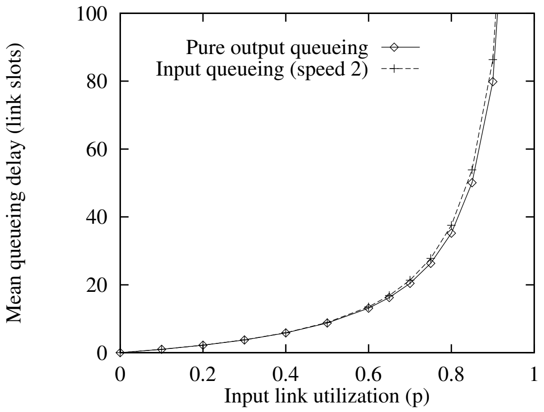

Fig. 3. The delay performance of an input queueing switch with a speedup factor of 2 is compared with the delay performance of pure output queueing. All the results are obtained using simulations. The arrival of packets on each input link is bursty with a mean burst size $b = 10$. The expected number of packets in each link slot equals p.

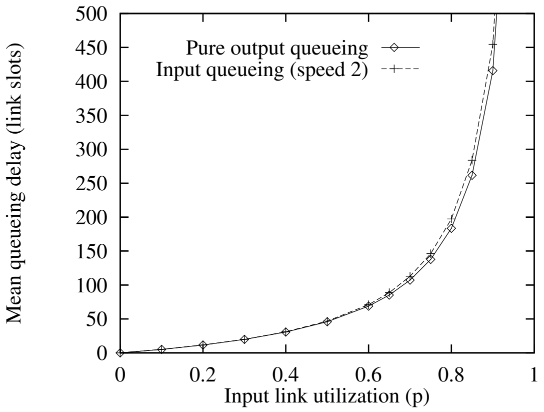

Fig. 4. The delay performance of an input queueing switch with a speedup factor of 2 is compared with the delay performance of pure output queueing. All the results are obtained using simulations. The arrival of packets on each input link is bursty with a mean burst size $b = 50$. The expected number of packets in each link slot equals p.

line blocking phenomenon.

We now consider bursty traffic with the model being as defined in Section II, with a mean burst size of $b = 10$ packets. The saturation throughput calculated from the result in [8] is 50.6%, for large n. For a finite switch size of $n = 16$, the saturation throughput can be expected to be a little higher. Nevertheless, by Proposition 1, a speedup factor of $2 > 1/0.506$ should be sufficient to achieve full input link utilization. The mean queueing delays achieved by output queueing and input queueing with a speedup factor of two are shown in Figure 3. It can be seen that the queueing delay performance is nearly identical. Note that since the saturation throughput for bursty traffic monotonically decreases with b and tends to 0.5 for large b, by Proposition 1, a speedup factor of 2 is sufficient to achieve full input link utilization for *all burst sizes.* For illustration, Figure 4 shows the mean queueing delays for an average burst size $b = 50$.

IV. Switch Fabrics with Memory

Another option to overcome the throughput limitations of input queueing, that has been investigated, is to use switch fabrics with memory [9]. To describe the operation of such fabrics, let us assume that a memory of m packets per output port is provided within the switch fabric. This means that as long as the number of packets waiting within the switch fabric for a specific output port, plus the number of head-of-line packets at the switch inputs for the same output port, is less than m, there is no head-of-line blocking. When there are m packets waiting for an output, new packets for that output wait in a queue at their respective input ports, blocking further transmission from that input port. This causes again the head-of-line blocking phenomenon, albeit in fewer cases than without the benefit of memory within the switch fabric. The performance of such a fabric approaches that of output queueing when m is large. Iliadis and Denzel [9] have shown that the saturation throughput of such a fabric increases with m and approaches unity for large m. They have derived an expression for the saturation throughput of such a switch, in the limit as $n \to \infty$, for the case of independent cell arrivals ($b = 1$), using which they show that the saturation throughput increases considerably over the value obtained for $m = 0$ (pure input queueing) even for moderate values for m. For example, the saturation throughput for $m = 8$ ($b = 1$, n large) is 0.8426.

Proposition 1 holds for fabrics with memory as well so that a speedup factor equal to the reciprocal of the saturation throughput is sufficient to achieve full input link

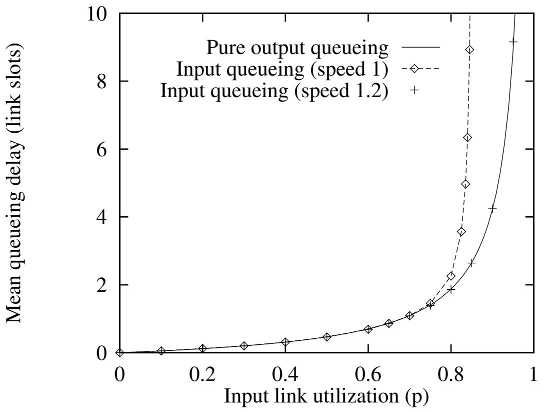

Fig. 5. The delay performance obtained by simulations of a switch fabric with a memory within the fabric of eight packets per output port, with a speedup factor of 1 (no speedup) and a speedup factor of 1.2, is compared with the delay performance of pure output queueing. The mean burst size $b = 1$ so that the destinations of successive packets are independent. The expected number of packets in each link slot equals p.

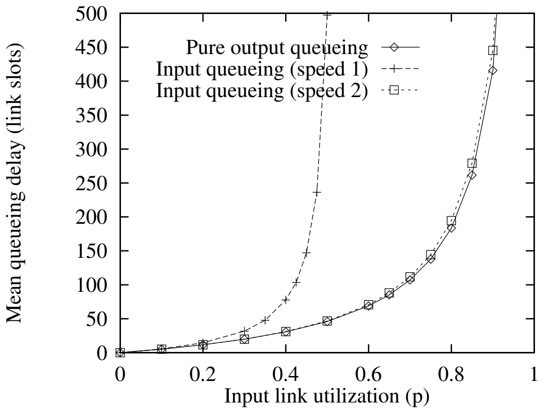

Fig. 6. The delay performance obtained by simulations of a switch fabric with a memory within the fabric of eight packets per output port, with a speedup factor of 1 (no speedup) and a speedup factor of 2, is compared with the delay performance of pure output queueing. All the results are obtained using simulations. The arrival of packets on each input link is bursty with a mean burst size $b = 50$. The expected number of packets in each link slot equals p.

utilization. Since this saturation throughput is closer to unity even for moderate values of m, a small amount of memory within the fabric can significantly reduce the speedup factor needed to achieve full input link utilization. For example, for $m = 8$, a speedup factor of $1.2 > 1/0.8426$ is sufficient. In Figure 5, we have plotted the mean queueing delays for a switch fabric with $m = 8$ without speedup and a speedup factor of 1.2. The number of switch ports is $n = 16$ and the packet destinations are chosen independently ($b = 1$). The performance of output queueing is also plotted for comparison. It can be observed that not only the throughput, but also the mean queueing delay performance approaches that of output queueing, when the switch fabric is operated at a speedup factor of 1.2.

Using memory within the switch fabric can be expensive, especially if it is to be implemented using on-chip memory, e.g., [18]. Therefore, reducing the required memory size by speeding up the fabric, even if only modestly, can represent a desirable design trade-off, even if it also increases the required memory speed and hence its cost. Another possibility to further reduce the total amount of memory and/or speedup factor necessary to achieve the desired level of performance, is sharing of the available memory within the fabric among the switch outputs; see [18] again for a sample design.

The above benefits notwithstanding, it should however be noted that the amount of memory required to achieve a given level of performance increases with the burst size. For example, the amount of memory that is sufficient to significantly increase the saturation throughput for $b = 1$, is quite inadequate for even moderately bursty traffic. Intuitively, this is so because in this case, a single burst of packets can fill up all the memory dedicated for a particular output and cause head-of-line blocking. When the mean burst size is much larger than the memory per output port, the performance approaches that of a fabric with no memory and the required speedup factor to achieve full input link utilization approaches two. This phenomenon is illustrated in Figure 6 where we compare the mean queueing delays achieved without speedup and with a speedup factor of 2, with the mean queueing delay achieved by output queueing for burst traffic with mean burst size $b = 50$ cells.

V. Conclusions

In this note, we investigated the impact of speedup on the performance of "input queueing" packet switches, and highlighted the fact that only a moderate speedup factor (at most two) is necessary to approach the delay and throughput performance of pure output queueing switches. This result, while not entirely unknown, seems to have been largely over-looked. Given its practical significance for the design and operation of high speed switch fabrics, especially with the continued increase in link speeds, we hope this note will help remedy this situation.

References

[1] H. Ahmadi and W. E. Denzel, "A survey of modern high-performance switching techniques," *IEEE Journal of Selected Areas in Communications*, vol. 7, no. 7, pp. 1091–1103, Sep. 1989.

[2] I. Cidon, I. Gopal, G. Grover, and M. Sidi, "Real-time packet switching: A performance analysis," *IEEE Journal of Selected Areas in Communications*, vol. 6, no. 9, pp. 1576–1586, Dec. 1988.

[3] I. Cidon, I. Gopal, P. M. Gopal, R. Guérin, J. Janniello, and M. Kaplan,

"The plaNET/ORBIT high speed network," *Journal of High Speed Networks*, vol. 2, no. 3, pp. 1–38, September 1993.

[4] N. McKeown, V. Anantharam, and J. Walrand, "Achieving 100% throughput in and input-queued switch," in *Proc. IEEE INFOCOM'96*, San Francisco, CA, March 1996, pp. 296–302.

[5] N. McKeown, Martin Izzard, Adisak Mekkittikul, Bill Ellersick, and Mark Horowitz, "The Tiny Tera: A packet switch core," in *Proc. Hot Interconnects V*, Stanford University, CA, August 1996.

[6] M. Karol, M. Hluchyj, and S. Morgan, "Input versus output queueing on a space division switch," *IEEE Transactions on Communications*, vol. 35, pp. 1347–1356, 1987.

[7] J. Y. Hui and E. Arthurs, "A broadband packet switch for integrated transport," *IEEE Journal of Selected Areas in Communications*, vol. 5, no. 8, pp. 1264–1273, Oct. 1987.

[8] S.-Q. Li, "Performance of a non-blocking space-division packet switch with correlated input traffic," in *Proc. IEEE GLOBECOM*, 1989, pp. 1754–1763.

[9] I. Iliadis and W. E. Denzel, "Analysis of packet switches with input and output queueing," *IEEE Transactions on Communications*, vol. 41, no. 5, pp. 731–740, May 1993.

[10] L. Jacob and A. Kumar, "Saturation throughput analysis of an input queueing ATM switch with multiclass bursty traffic," *IEEE Transactions on Communications*, vol. 43, no. 2/3/4, Feb./Mar./Apr. 1995.

[11] L. Jacob and A. Kumar, "Delay performance of some scheduling strategies in an input queueing ATM switch with multiclass bursty traffic," *IEEE/ACM Transactions on Networking*, vol. 4, no. 2, pp. 258–271, Apr. 1996.

[12] A. Kumar and L. Jacob, "A stability condition for the input buffers of an input queueing ATM cell switch," Manuscript under review.

[13] Y.-S. Yeh, M. G. Hluchyj, and A. S. Acampora, "The Knockout switch: A simple, modular architecture for high-performance packet switching," *IEEE Journal of Selected Areas in Communications*, vol. 5, no. 8, pp. 1274–1283, Oct. 1987.

[14] Y. Oie et al., "Effect of speedup in nonblocking packet switch," in *Proc. IEEE ICC*, 1989, pp. 410–414.

[15] J. S.-C. Chen and T. E. Stern, "Throughput analysis, optimal buffer allocation, and traffic imbalance study of a generic nonblocking packet switch," *IEEE Journal of Selected Areas in Communications*, vol. 9, no. 3, pp. 439–449, Apr. 1991.

[16] A. Pattavina and G. Bruzzi, "Analysis of input and output queueing for nonblocking ATM switches," *IEEE/ACM Transactions on Networking*, vol. 1, no. 3, pp. 314–328, Jun. 1993.

[17] S. C. Liew, "Performance of various input-buffered and output-buffered ATM switch design principles under bursty traffic: Simulation study," *IEEE Transactions on Communications*, vol. 42, no. 2/3/4, pp. 1371–1379, Feb./Mar./Apr. 1994.

[18] W. E. Denzel, A. P. J. Engbersen, and I. Iliadis, "A flexible shared buffer switch for ATM at Gb/s rates," *Computer Networks and ISDN Systems*, vol. 27, no. 4, pp. 611–624, January 1995.