Journal of High Speed Networks 10 (2001) 161–186
IOS Press

Performance analysis of speeded-up high-speed packet switches [1]

Aniruddha S. Diwan [a,*], Roch Guérin [b] and Kumar N. Sivarajan [c]

[a] *ECE Department, Indian Institute of Science, C. V. Raman Avenue, Bangalore 560 012, India*
E-mail: diwan@ece.iisc.ernet.in
[b] *University of Pennsylvania, Department Electrical Engineering, Rm. 367 GRW, 200 South 33rd Street, Philadelphia, PA 19104, USA*
E-mail: guerin@ee.upenn.edu
[c] *Tejas Networks, Khanija Bhavan, Race Course Road, Bangalore 560 001, India*
E-mail: kumar@tejasnetworks.com

Abstract. In this paper, we study the performance of high-speed packet switches, where the switch fabric operates at a slightly higher speed than the links, i.e., a speeded-up switch. Such structures are by no means new and there are two well studied architectures in the literature for such packet switches: pure input queueing (no speedup) and pure output queueing (speedup of N, the number of links), with output queueing switches offering substantial performance benefits. However, as link speeds keep increasing, the speedup of N needed for pure output queueing becomes a significant technical challenge. This is one of the main reasons for the renewed interest in *moderately speeded-up switch fabrics*. The aim of this paper is to highlight the result that only a moderate speed-up factor (less than two) is sufficient to achieve full input link utilization. In particular, we emphasize that this holds, even without relying on a central switch controller making intelligent decisions on which packets to schedule through the switch. As shown in recent works, i.e., [5, 17, 20, 23, 25] there are clearly benefits to using intelligent controllers, but they do come at a cost. Instead, in this paper we focus on what can be achieved by relying simply on switch speedup. We do so by means of analysis and simulations. Our analysis provides explicit expressions for the average queue length in switches with integer and rational speedups. The results are complemented by simulations that are used to obtain delay estimates, and which also allow us to extend our investigation to shared memory switches for which we find that good performance can be achieved with an even lower speedup.

Keywords: Input queueing, output queueing, high-speed packet switches, speedup factor, HOL blocking

1. Introduction

1.1. Basic switch architecture

A *packet switch* consists of input and output ports (or links) connected to a switch fabric. The function of a packet switch is to transport packets from the input ports to the appropriate output ports. Two classes of packet switches arise in communication networks namely, *blocking* and *nonblocking* packet switches. In a blocking packet switch, a packet going to some output port may be blocked because of contention inside the switch with packets destined for other output ports. On the other hand, a nonblocking packet switch avoids all contentions inside the switch for packets destined to different output ports. In both blocking and non-blocking switches, contention remains inevitable among packets addressed to the same output port. In this paper, we limit ourselves to nonblocking packet switches.

Consider the packet switch of Fig. 1. This switch has N input ports and N output ports and employs a combination of input queueing and output queueing. Fixed-length packets, or cells, arrive at the input ports of the packet

[1] A part of this work was done when Roch Guérin and Kumar N. Sivarajan were at IBM T. J. Watson Res. Ctr. A summary of this paper has appeared in *Proceedings of the IFIP Fifth Int. Conf. on Broadband Comm. '99.*

* Corresponding author.

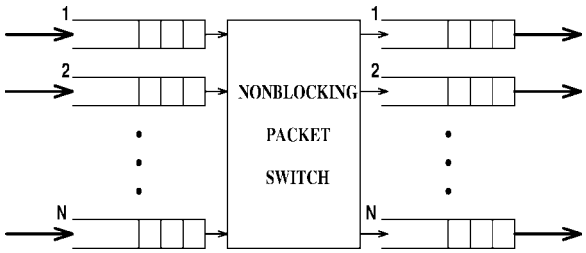


Fig. 1. An $N \times N$ speeded-up packet switch with input and output queues.

switch. Note that this cell structure is only internal to the switch, so that the links could carry variable size packets. However, in the rest of the paper, we focus on the case of fixed-size packets. Each packet contains an identifier that indicates which output[2] j, $1 \leqslant j \leqslant N$, it is destined for. In this paper, we are concerned with the performance analysis of such a switch fabric when it is *speeded-up*. That is, the switch fabric runs at a speed greater than that of the input and output links. For simplicity, we assume in our analysis that all input and output links run at the same speed.

Another important characteristic of the switch we consider is that each input makes independent decisions on which packet to send, i.e., there is no coordination amongst inputs aimed at selecting the 'best' combination of packet destinations in a given time slot. As a result, the queueing structure at the inputs is a simple FIFO queue, with the first packet in the queue attempting to go through the switch. Such a structure can be extended to support priorities, i.e., have separate queues for each priority class, but this still preserves the lack of destination awareness at each input, when deciding which packet to send through the switch. This is in contrast to the switch fabrics of [5,17,20,23,25], which assume that packets are sorted according to their destination, and possibly priority, at the inputs, so that a central controller can make a selection based on the best possible combination of packets to send through the switch.

The operation of the simple switch fabric which we consider assumes, therefore, independent transmission attempts from all inputs. The basic packet transmission time through the switch fabric is called a *switch-slot*, and switch-slot boundaries at all the links are synchronized. For our analysis, we assume that packet arrivals on all N links are statistically identical, and packet destinations are independent from packet to packet and uniformly distributed across all N outputs. We denote as a *link-slot* the time it takes for a packet to arrive on the link. We also assume that the statistics of packet arrivals on all input links are identical. Because of potential contentions, it may take several switch slots to transfer a packet through the switch, as only packets with distinct destinations can be transferred within a given switch-slot. However, the impact on the inputs depends on the relation between switch slots and link slots, or in other words on the speedup s of the switch. For example, an output queueing switch corresponds to a switch slot that is N times shorter than a link slot, so that independent of the distribution of destinations, all packets can be transferred through the switch before the start of the next link slot. In switches with a speedup less than N, some queueing will occur on the inputs, and in our analysis we assume infinite buffers on inputs (and outputs), so that no packets are lost. In cases when packets from multiple inputs contend for the same destination in a switch-slot, one packet is chosen according to a *contention resolution policy*, e.g., round-robin or random, and other contending packets wait in their input queue.

1.2. Previous works and motivations

The two extremes, namely pure input queueing and pure output queueing, of the switch fabric we consider in the paper have been well studied in the literature. In a pure input queueing switch, the switch fabric runs at the same speed as that of the input and output links, and as a result such fabrics suffer from what has been called *Head-Of-Line (HOL) blocking* [11,16]. This occurs when packets destined to an output currently free of contention are

[2] In the paper, we limit ourselves to the case of unicast flows.

blocked because the first packet in their input queue is contending with packets from other inputs and headed to the same destination. Because of this phenomenon, the saturation throughput of this switch architecture is limited. Hui and Arthurs [11] and Karol, Hluchyj and Morgan in their seminal paper on input versus output queueing switches [16] have shown that the saturation throughput of pure input queueing switches for uniform i.i.d. Bernoulli arrivals at inputs, is $2 - \sqrt{2} \approx 0.586$, for large N. Further results were obtained by Li [18], who showed that the saturation throughput of pure input-queued switches for the case of bursty traffic tends to 0.5 for large mean burst size and for large N. In contrast, an $N \times N$ pure output queueing switch does not suffer any such throughput limitation. As mentioned before, such a switch fabric can always transfer within each link slot, all the packets arriving at each input. As a result, such a switch achieves a saturation throughput of 100% (of the link speed).

Thus the performance of pure output queueing switches is clearly superior, but this comes at the cost of a much higher speed for the switch fabric. As link speeds keep increasing, it becomes increasingly challenging to develop switch fabrics with significant speedups. As a result, it is desirable to better understand the exact trade-off between performance and speedup. There has been a number of works on moderately speeded-up switch architectures, whose performance can approximate that of pure output queueing switches. One of the earlier proposal was the Knockout Switch of [27]. In the Knockout architecture, up to $L, 1 \leqslant L \leqslant N$, HOL packets can be transported to each output in a time slot. Thus, $L = 1$ leads to a pure input queueing switch, whereas $L = N$ leads to a pure output queueing switch architecture. L is called the *parallelism* factor. Oie et al. [22] and Chen et al. [6] have separately shown that as L increases, the throughput increases rapidly and for $L = 4$, it is in excess of 99%. Li [18] also analyzed this and came up with the same result.

While these results showed that near 100% could be achieved with a relatively moderate speedup, they have also led to the general belief that a speedup of N was required to get full 100% throughput. However, this is not so, and this is due to the fact that there is a subtle distinction between parallelism and speedup. For example, in the case of a parallelism of $L = 2$, all the packets that are switched within a link slot must be from *distinct* input links. On the other hand, a speed-up of 2 does not impose such a constraint, and two packets from the same input can be switched during one link slot. This seemingly innocuous difference in the operation of the switch fabric, therefore, has reasonably significant consequences. This had been recognized or hinted to in a few early papers. In particular, [11] mentioned that the use of parallel switch planes can allow full link utilization, provided the operation of the two planes is staggered from each other by half a slot. However, probably because this result was not the main focus of the paper, this statement seems to have been overlooked by most. Similarly, Liew [19] observed through simulations that a speed-up of 2 was sufficient to ensure a throughput of 100%. But again, because his focus was on packet loss probabilities, this result was mostly overlooked. As a result, one of the goals of this paper is to ensure that this result is publicized, as well as provide further evidences of the fact that only a moderate speedup is required to closely approximate the performance of pure output queueing switches.

Before we proceed with our analysis, it is important to position and contrast this work, and in particular, the switch structure it assumes, against a number of recent proposals for new input queueing switch architectures. These architectures share the motivation of closely emulating the performance of output queueing switches using input queueing switches with no or minimal speedup. The first work in that space was that of McKeown et al., who showed in [20] that 100% throughput could be achieved with a pure input queueing switch. Since then, there have been several recent works which have extended the throughput results of [20] to stronger results, that give the speedup required to *exactly* emulate the behavior of an output queueing switch [5,17,23,25].

While these works certainly represent an important breakthrough in the design of switch architecture, it is important to realize that they entail a certain cost that can be substantial. In particular, all these results assume that the switch input adapters demultiplex packets headed to different destinations (or corresponding to individual flows, if per flow guarantees are required) into different queues, and make available to a central switch controller the state of the different queues they maintain. Typically, this information needs to be provided to the central controller in each switch slot, where it is used as input to a *matching* algorithm that determines the 'best' pattern of packets to send through the switch. The complexity, and therefore cost, of such a structure is in both the number of queues needed in each adapter, and the central controller which must be capable of performing a reasonably complex matching task in each time slot. There are clearly optimizations and trade-offs that are possible, but in contrast, in the switch

fabric which we assume, the inputs operate independently of each other and only need a single queue. The performance guarantees such simpler fabrics can provide are clearly not as strong, but given a sufficient speedup, they may be adequate in some environments. A better understanding of this potential is the main goal of this paper.

1.3. Outline of the paper

In Section 2 we study the throughput of speeded-up switch fabrics. In Section 3 we analyze queue sizes in switch fabrics with speed-up factors of the form r/s. We use a traffic model which is a suitable adaptation of the uniform i.i.d. Bernoulli arrival model. Input queues and output queues are analyzed separately. An exact model is developed for input queues whereas an approximate model is developed for output queues. The analysis developed here is general in the sense that it is applicable to any speed-up factor. In Section 4 we compare the analysis with the simulations. Various results are shown for three cases: speed-up factors of 1.5, 2 and 3. In Section 5 we present some simulation results regarding packet delays in moderately speeded-up switch fabrics. Three cases are described: packet delays for uniform i.i.d. traffic, packet delays for bursty traffic and packet delays in a speeded-up switch fabric with memory.

2. Switch throughput

Let us denote the transmission time of a packet on any of the links by $1/R_l$ and the transmission time of a packet inside the switch by $1/R_s$. Thus R_s and R_l are the transmission rates (in packets per second) of the switch and input links, respectively. When the switch fabric is speeded up, $R_l < R_s$, and the ratio R_s/R_l is termed the *speed-up factor*. 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 $q = (R_l/R_s)p$. We say that a switch input is *saturated* if it has a packet available for transmission through the switch fabric, in every switch-slot. 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 and the speed-up factor. At one extreme, for a speed-up factor of 1, if all packets on all input links are destined to a single output port i, the saturation throughput is $1/N$. When packets on input link $i, 1 \leqslant i \leqslant N$, 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 in Section 1.1, 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.)

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. The packets that cannot be transmitted through the switch immediately upon arrival, due to HOL blocking, are queued in an infinite buffer at each input link. We define the *stability throughput* as the maximum link utilization 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 queues are stable [12]. However, that this is an assumption requiring proof has been recognized by Jacob and Kumar [14,15] and their proof for the case of Bernoulli arrivals (successive packets have independent destinations) with identical rates on all input links may be found in [13].

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 is less than the saturation throughput of the switch. We will assume that the packet arrival processes arising in our discussion satisfy the Saturation-Stability property.

We now 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, 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 the 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 $q = (R_l/R_s)p$. 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%. □

Consider a nonblocking switch fabric of size $N \times N$. As N grows large, the saturation throughput of this switch fabric is 0.586 (of switch speed) for the uniform i.i.d. Bernoulli arrival model [16] and tends to 0.5 (of switch speed) for bursty arrivals [18]. [3] Hence for this switch fabric, $\gamma = 0.5$. In the following discussion, we consider the bursty arrival process of [18] and we assume that this arrival process satisfies the Saturation-Stability property.

First we analyze the stability throughput of pure input queued switches. We want to show that a pure input queued switch has a stability throughput of only 50% for the above input process assumptions. Note that, in a pure input queued switch, it takes a full link-slot to transfer a packet across the switch fabric and a switch-slot is equivalent to a link-slot. Hence $R_s/R_l = 1$ and $q = p$. According to the Saturation-Stability assumption, as long as $q < \gamma$, the queues are stable which means $p < 0.5$. Thus the maximum input link utilization for a pure output queued switch is only 50%. The implication of this is we cannot send packets at a rate > 0.5 (per link-slot) on the input ports of the switch fabric as otherwise the input queues become unstable. Such behavior is clearly undesirable as we do not want to limit the packet arrival rate on the input ports.

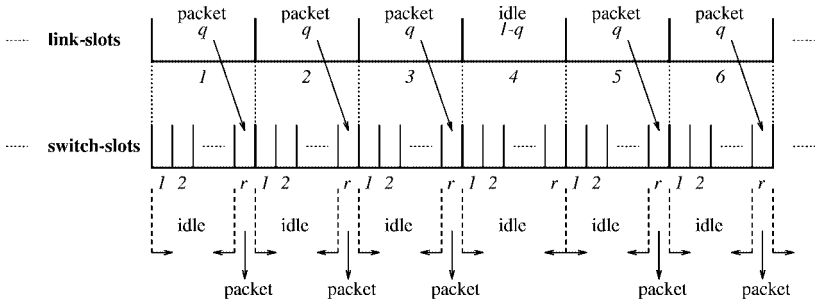
We now analyze the stability throughput of pure output queued switches. We want to show that a pure output queued switch has a stability throughput of 100%. Note that for a pure output queued switch, there are N switch-slots in a link-slot and it takes 1 switch-slot to transfer a packet across the switch. Thus $R_s/R_l = N$ and $q = p/N$. According to the Saturation-Stability assumption, as long as $q < \gamma$ the queues are stable, which means $p/N < 0.5$. This is true for $p \leqslant 1$. Thus the maximum input link utilization for a pure output queued switch is 100%. What this means is that even if every link-slot has a packet on an input link, the input queues are stable. Thus the performance of pure output queueing is clearly superior.

We now apply Proposition 1 to the nonblocking switch fabric to obtain the minimum speedup factor that is required to achieve 100% stability throughput. That is, given $\gamma = 0.5$ and $p = 1$, what is the minimum value of R_s/R_l? Clearly, the speedup factor is between 1 and N. From proposition 1, $R_s > R_l/0.5$, or, $R_s/R_l > 2$. Thus, a speedup factor of 2 can achieve 100% input link utilization. In Section 5.3, we will observe that much less speedup is required for switch fabrics with memory.

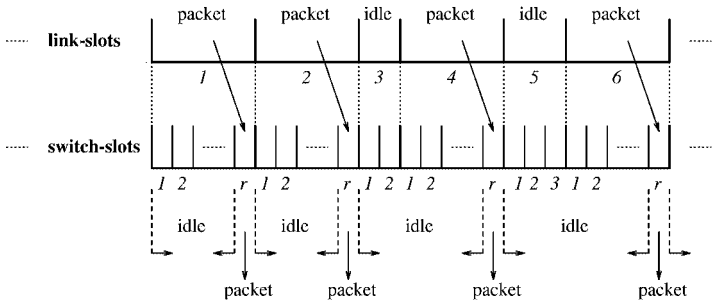
3. Queue length analysis

In this section we analyze the queue lengths at various queueing points in a speeded-up switch fabric. The time it takes to transmit a packet fully on any link is called a *link-slot*. A link-slot consists of r switch-slots. The time it takes to transfer a packet from an input port to an output port, i.e., inside the switch fabric, is s switch-slots. In other words, the link speed is less by a factor of r/s than the switch fabric speed and we say that the speed-up factor of the switch fabric is r/s. Note that this type of model is applicable to any speed-up factor of the form r/s and is thus a general model. It can be seen that pure input-queued switches ($r = s = 1$) and pure output-queued switches ($r = N, s = 1$) are also special cases of the above model.

[3] For small values of N saturation throughput is higher.



(a) synchronous link-slots and synchronous switch-slots



(b) asynchronous link-slots and synchronous switch-slots

Fig. 2. Adaptation of uniform i.i.d. Bernoulli arrival process.

3.1. Packet arrival process

In practice, it is very difficult to theoretically characterize the exact nature of packet arrivals because of the presence of diverse traffic classes in high-speed networks. Hence most of the work on packet switching considers arrival processes which make the analysis theoretically tractable and reasonably approximate the real life scenario. Uniform i.i.d. Bernoulli arrival process is one which makes analysis theoretically tractable. We now describe the packet arrival process which is considered in this paper for analyzing the speeded-up switch fabric. This arrival process is a suitable adaptation of the uniform i.i.d. Bernoulli arrival process. In the uniform i.i.d. Bernoulli model a packet arrives at an input port with probability q in a link-slot and its destination output port is chosen uniformly randomly among all the N output ports. The arrival of a packet in a given link-slot and at a given input port is statistically independent of the packet arrivals in any previous link slots and at any input port, including itself. Note that this arrival process is such that packets arrive in synchronized link-slots. In practice this need not be the case. That is, the next packet may arrive after some time which need not be in units of a link-slot. Further, for analytical simplicity, an arrival process over synchronized switch-slots is really required. For these reasons, we suitably adapt the above mentioned uniform i.i.d. Bernoulli process as follows:

Consider a typical sequence of packet arrivals at a given input port over the link-slots as illustrated in Fig. 2. A packet is wholly available to the switch only when the last bit of that packet has arrived at the input port. It takes one whole link-slot or r switch-slots for all the bits of a packet to arrive. As far as the switch-slots are concerned, it can be said that packets arrive in switch-slots and there are always $(r - 1)$ idle switch-slots before a packet arrival in a switch-slot. Therefore a packet arrival in a link-slot is equivalent to $(r - 1)$ idle switch-slots and a packet arrival in the switch-slot immediately following the idle switch-slots. An idle link-slot corresponds to r idle switch-slots. This is shown in part (a) of Fig. 2. Observe that even if we consider such a packet arrival model

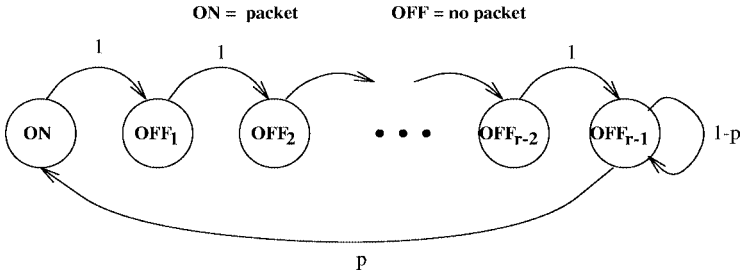


Fig. 3. Markov chain modeling the arrival process in the input queues of the packet switch.

over switch-slots, we still have synchronized link-slots. We can get rid of these synchronized link-slots by slightly relaxing the restriction of one whole idle link-slot when there is no arrival. That is, there must be *at least* $(r-1)$ (and not necessarily exactly $(r-1)$, or $(r-1)$ plus an integer multiple of r) idle switch-slots between successive arrivals. This is shown in part (b) of Fig. 2.

Such a switch-slot arrival process leads to the Markov chain of Fig. 3. An ON switch-slot corresponds to an arrival. An OFF switch-slot corresponds to no arrival. The state transition probabilities are given in Fig. 3. The destination output port of a packet is chosen uniformly randomly out of N output ports. This is the arrival process at each input port of the switch fabric. The packet arrival rate, λ, in packets per switch-slot can be obtained by solving for the steady-state vector of this Markov chain and we get,

$$\lambda = \frac{p}{1+(r-1)p}, \tag{1}$$

and the rate of packet arrivals in packets per link-slot is $r\lambda$.

3.2. Asymptotic analysis of switch performance

The packet switch with the input and output queues forms a complex queueing network. It seems that the exact queueing analysis of this complex queueing network is intractable for finite N. In order to get a handle on the problem, we follow the approach outlined in [16], and try to analyze the different parts of the network separately. For this, consider the travelogue of a tagged packet from the moment it arrived at an input queue to the moment it is transmitted fully on its destination output link. The tagged packet encounters queueing at three points in the network. It is first queued in the input queue where it has arrived. The packets in each input queue are transfered on a FIFO (first in first out) basis. When all the packets in front of this tagged packet in its input queue are transmitted across the switch, the tagged packet enters the HOL position of this input queue. At this point, there may be packets in the HOL position of other input queues whose destination is the same as that of the tagged packet. There may also be a packet which is already in the transfer process to the destination of the tagged packet, in which case the destination is busy. The switch can start transferring only one of the former type of packets in a switch slot to the destination *only if* the destination is free. This is called *HOL contention*. This is the second queueing point in the network. Note that the HOL packets destined for a given output form a contention (or virtual) queue corresponding to that output. Therefore the tagged packet is queued in the HOL position, i.e., in its contention queue. Which packet from a given contention queue is to be transferred to the corresponding output (if it is free) in the next s switch-slots is decided by a *contention resolution policy*. The third queueing occurs in the output queue because the tagged packet is transmitted on the output link only when all the packets that arrived before it are transmitted fully on the output link.

We analyze the queue-length of these three types of queues for infinite input and output queues, and for the asymptotic case ($N \to \infty$) in the following subsections. Contention at the HOL positions is resolved thus: If k HOL packets contend for a particular output in a switch-slot, one of the k packets is chosen uniformly randomly

from those k packets. The other packets have to wait until that switch-slot in which the output becomes free and a new selection is made among the packets that are then waiting.

The arrival process of Section 3.1 is assumed to satisfy the saturation-stability property of Section 2. For this, the arrival rate, λ, should be less than the saturation throughput of the switch. Observe that a switch with only input queues is exactly the same as a pure input-queued switch on a switch-slot basis. The saturation throughput is 0.586 for the uniform i.i.d. Bernoulli arrival model [16] and tends to 0.5 for bursty arrivals [18] over the switch-slots. Therefore as long as $\lambda < 0.5$, the switch input queues are stable and, due to the symmetry among the output links and the equality of the transmission speeds on the input and output links, the switch output queues are also stable.

3.2.1. Contention queue analysis

The destination output of a packet is selected uniformly randomly among the N outputs as mentioned before. The situation at the HOL positions of the input queues is similar to that at the HOL positions of a pure input-queued switch. That is, in both cases packets encounter the HOL blocking phenomenon. We assume that each output contention process tends to be independent asymptotically (as $N \to \infty$). This is indeed the case for a pure input-queued switch with uniform i.i.d. Bernoulli traffic [16]. With this assumption the contention queues can be analyzed separately.

Consider one such contention queue, say corresponding to output i. In a given switch-slot, say mth, the contention queue contains backlogged as well as unbacklogged packets. A packet whose transfer is in progress in the mth switch-slot is also considered as a backlogged packet. We denote the number of backlogged packets by B_m^i and the number of unbacklogged packets by A_m^i. An input queue is unbacklogged, if and only if, either a packet transfer ended in the $(m-1)$th switch-slot or it was empty during the $(m-1)$th switch-slot. It then follows that,

$$B_m^i = \begin{cases} \max(0, B_{m-1}^i + A_m^i - 1) & : \text{if destination is free in } m\text{th switch-slot,} \\ \max(0, B_{m-1}^i + A_m^i) & : \text{if destination is busy in } m\text{th switch-slot,} \end{cases} \tag{2}$$

A_m^i, the number of packet arrivals during the mth switch-slot at the unbacklogged input queues and destined for output i, has the binomial probabilities

$$\Pr[A_m^i = k] = \binom{F_{m-1}}{k} (1/N)^k (1 - 1/N)^{F_{m-1}-k}, \quad k = 0, 1, \ldots, F_{m-1}, \tag{3}$$

where,

$$F_{m-1} = N - \sum_{i=1}^{N} B_{m-1}^i - (\text{ number of empty input queues}). \tag{4}$$

F_{m-1} is the number of unbacklogged nonempty input queues at the end of the $(m-1)$st switch-slot. F_{m-1} is also the number of input queues with new packets at their HOL positions during the mth switch-slot. ρ_0 is the utilization of the output links (i.e., the switch throughput). When the switch operates below saturation and the arrival process is assumed to satisfy the saturation-stability property (and we assume these conditions are satisfied), $\rho_0 = \lambda$. The binomial form of (3) leads to the assumption that, as $N \to \infty$, the steady-state number of packets moving to the head of unbacklogged input queues in each switch-slot, and destined for output i, (A^i), becomes Poisson at rate,

$$\overline{F}/N = \rho_0,$$

where $\overline{F}$ is the mean steady-state number of new packets at the HOL positions. In other words,

$$\lim_{N \to \infty} \Pr\{A^i = k\} = e^{-\rho_0} {\rho_0}^k / k\,!$$

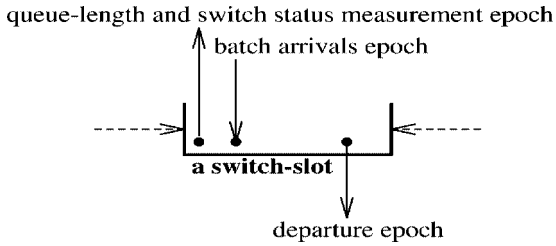


Fig. 4. Arrival, departure and measurement epochs for the analysis of contention queue.

and $\rho_0 = \lambda$ below saturation. That this 'Poisson process' assumption is correct, is substantiated by the simulated performance [8].

The above Poisson process assumption and the form of (2) suggest that the contention queue can be modeled as a discrete-time $BP/D_s/1$ queue with random order of service. BP represents discrete-time batch-Poisson process. D_s represents deterministic service time which, in this case, is s switch-slots (the transmission time of a packet inside the switch). (B_m^i) is the system queue-length of the contention queue for output i during the mth slot. This queueing model can be analyzed to get the steady-state queue-length distribution in a random switch-slot and the delay distribution of the packets in the queue. Each output contention process is assumed to be independent as $N \to \infty$ and all the output contention processes are identical. Hence it is sufficient to analyze any one of them.

Note that when the speed-up factor is of the form $r/1$, i.e., integer speed-up factor r, the contention queue is modeled as $BP/D_1/1$. This is the same queueing model as that of the contention queue in [16]. There this model is referred to as discrete $M/D/1$ queue. Thus the analysis of this discrete $M/D/1$ developed in [16] is directly applicable to the case of *integer* speed-up factors. The analysis of the contention queue in [16] is a special case of the analysis developed below.

3.2.1.1. Distribution of queue-length in a random switch-slot. We cannot model the queue-length of the contention queue *alone* by a DTMC. If we know how many switch-slots of service time have been completed for the current packet in transfer out of the deterministic s switch-slots of service time, we can model the queue-length by a DTMC. The switch status is T_0 at the beginning of a switch-slot if the switch is not busy at a switch-slot boundary. In this status the switch is ready to serve any packet in this switch-slot. If a packet arrives when the status is T_0 and the contention queue is empty, the packet starts service immediately from this switch-slot. If say, n packets are in the contention queue waiting for service, one of them starts service from this switch-slot according to the contention resolution policy if no packet arrives in this switch-slot. If a packet arrives in this switch-slot, one of the total $(n+1)$ packets starts service from this switch-slot. The status T_i, where $1 \leqslant i < s$, at the beginning of a switch-slot indicates that there is a packet transfer in progress and that first i switch-slots of service have been completed. The precise epochs of arrivals (if any), departure (if any) and the queue-length as well as the switch status measurements in a switch-slot are shown in Fig. 4. A packet is eligible to get service in the same switch-slot in which it arrives.

We club the queue-length in a switch-slot and the switch-status of that switch-slot to form a state. This state can be modeled by a 2-D DTMC over the state space $\{(i,j) : i \geqslant 0, j \in \{T_0, T_1, \ldots, T_{s-1}\}\}$. Then queue-lengths in successive switch-slots follow this 2-D DTMC. It is assumed that the number of arrivals in the contention queue during each switch-slot has Poisson probabilities,

$$a_i = \Pr\{A = i\} = \frac{\lambda^i e^{-\lambda}}{i!}, \quad i = 0, 1, 2, \ldots. \tag{5}$$

Therefore the state transition probability matrix takes the form,

$$\mathbf{P} = \begin{bmatrix} B_0 & B_1 & B_2 & B_3 & \cdots \\ C & A_1 & A_2 & A_3 & \cdots \\ \tilde{0} & A_0 & A_1 & A_2 & \cdots \\ \tilde{0} & \tilde{0} & A_0 & A_1 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}, \tag{6}$$

where B_0 is an 1×1 matrix, $B_i, i > 0$ are $1 \times s$ matrices, C is an $s \times 1$ matrix and $A_i, i \geqslant 0$ are $s \times s$ matrices. We have ordered the switch-status as follows : $T_i = i$ for all the sub-matrices of **P**. E.g., an entry $a_{j,k}$ in an A_i matrix represents a status transition from status j to status k. B_0 has only one entry which is $b_{0,0}$ which means the switch-status of the 2-D DTMC before transition is 0 and after transition it becomes 0. The row and column of a sub-matrix entry in **P** give the queue-length before and after transition respectively, e.g., B_0 is at row 0 and column 0, so for B_0 the queue-length before transition is 0 (row) and the queue-length after transition is 0 (column).
The B_0 matrix is as follows,

$$B_0 = [a_0] = \left[e^{-\lambda}\right]. \tag{7}$$

The C matrix is as follows,

$$C = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ a_0 \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ e^{-\lambda} \end{bmatrix}. \tag{8}$$

The A_0 matrix is as follows,

$$A_0 = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ a_0 & 0 & \cdots & 0 \end{bmatrix}, \tag{9}$$

where $a_0 = e^{-\lambda}$.
The $B_i, i \geqslant 1$ matrices are as follows,

$$B_i = [\,0 \quad a_i \quad 0 \quad \cdots \quad 0\,], \tag{10}$$

and the $A_i, i \geqslant 1$ matrices are as follows,

$$A_i = \begin{bmatrix} 0 & a_{i-1} & 0 & \cdots & 0 \\ 0 & 0 & a_{i-1} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_{i-1} \\ a_i & 0 & \cdots & 0 & 0 \end{bmatrix}, \tag{11}$$

where a_i are given by (5). Note that the **P** matrix is a structured $M/G/1$-type of matrix [21]. Efficient matrix-analytic methods have recently been developed in queueing theory [2–4] which can be employed to get the steady-state probabilities vector of such structured matrices of $M/G/1$ type. Some of these methods have been

implemented in a software package called TELPACK [1]. We have used TELPACK to compute the steady-state probability vector of the **P** matrix. The steady-state probability vector of the **P** matrix is of the form,

$$\pi = \left[\pi^0\ \pi^1\ \pi^2\ \cdots\ \pi^i\ \cdots\right], \tag{12}$$

where $\pi^0 = [q_{0,\,T_0}]$ and,

$$\pi^i = \left[q_{i,T_0}\ q_{i,T_1}\ \cdots\ q_{i,T_{(s-1)}}\right], \quad : i > 0, \tag{13}$$

q_{i,T_j} is the steady-state probability of the queue-length being i and the switch status being T_j in a random switch-slot. We can now compute the steady state probability vector of queue-length in a random switch-slot which is,

$$\tilde{l} = [l_0\ l_1\ l_2\ \cdots\ l_i\ \cdots], \tag{14}$$

where l_i is the probability that the queue-length is i in a random switch-slot. The l_is are given by,

$$l_0 = q_{0,T_0}, \tag{15}$$

$$l_i = \sum_{j=0}^{s-1} q_{i,T_j}. \tag{16}$$

3.2.1.2. Distribution of delay. Karol et al. [16] have developed a simple numerical method for computing the delay distribution of a discrete-time $BP/D_1/1$ queue, with packets served in random order (Appendix III of [16]). In this section we extend this scheme to the case of a discrete-time $BP/D_s/1$ queue, with packets served in random order. The number of packet arrivals at the beginning of each switch-slot is Poisson distributed with rate λ and each packet requires s switch-slots of service time. We focus our attention on a particular 'tagged' packet in the system, during a given switch-slot. Let p^m_{k,T_0} denote the probability, conditioned on there being a total of k packets in the system and the switch status being T_0 during the given switch-slot, that the remaining delay is m switch-slots until the tagged packet completes service.
The p^m_{k,T_0} can be obtained by recursion on m as follows,

$$p^m_{1,T_0} = \begin{cases} 1 : m = s, \\ 0 : m \neq s, \end{cases} \tag{17}$$

$$p^s_{k,T_0} = \frac{1}{k} : k \geqslant 1, \tag{18}$$

$$p^m_{k,T_0} = (k-1)p^s_{k,T_0} \cdot \sum_{j=0}^{\infty} p^{m-s}_{k-1+j,T_0} \cdot \frac{e^{-s\lambda}(s\lambda)^j}{j\,!} : \begin{cases} m = x \cdot s, \\ x > 1, \\ k > 1, \end{cases} \tag{19}$$

and $p^m_{k,T_0} = 0$ if m is not a multiple of s.
Averaging over k, the delay D has probabilities as follows:
For $m = s \cdot i$ such that $i \geqslant 1$ and i integer,

$$\Pr\{D = m\} = \sum_{k=1}^{\infty} p^m_{k,T_0} \cdot \Pr\,[k \text{ packets in system immediately after the tagged packet arrives and status is } T_0 \text{ in the switch-slot in which the tagged packet arrives}]$$

$$= \sum_{k=1}^{\infty} p^m_{k,T_0} \cdot \sum_{n=0}^{k-1} q_{n,T_0} \cdot \frac{e^{-\lambda}\lambda^{k-n-1}}{(k-n-1)\,!}. \tag{20}$$

For $m = s \cdot i + j$ such that $i \geqslant 1$ and $1 \leqslant j \leqslant (s-1)$ and i, j integers,

$$\Pr\{D = m\} = \sum_{k=1}^{\infty} p_{k,T_0}^{m-j} \cdot \Pr[k \text{ packets in the system in the } j\text{th switch-slot after that in which the tagged packet arrives and status is } T_{(s-j)} \text{ in the switch-slot in which the tagged packet arrives}]$$

$$= \sum_{k=1}^{\infty} p_{k,T_0}^{m-j} \cdot \sum_{n=0}^{k-1} q_{n,T_{(s-j)}} \cdot \frac{e^{-(j+1)\lambda}((j+1)\lambda)^{k-n-1}}{(k-n-1)\,!}. \tag{21}$$

For all the remaining m, the delay probability is zero. The q_{i,T_j} are obtained from (12). With (20)–(21) we get the delay distribution of a packet in the system. The moments of the delay distribution are determined numerically from the delay probabilities in (20)–(21).

3.2.2. Input queue analysis

As $N \to \infty$, successive packets in an input queue i experience the same service time distribution because their destination addresses are independent and are equiprobable. Observe that the number of switch-slots elapsed between the entry of a tagged packet in the HOL position of its input queue and the exit of that packet from the input queue, is equal to the delay of the packet in the contention queue. It is as if the tagged packet is served for that many switch-slots in its input queue. In other words, the service time distribution of a packet in an input queue is the delay distribution of a packet in the contention queue. With the arrival process of Section 3.1 and the service time distribution of Section 3.2.1, we can model the input queue as a $G/G/1$ queue.

We analyze the queue-length of this queueing system as follows. Note that the queue-length in successive switch-slots cannot be modeled by a DTMC. Thus it is difficult to compute the steady-state distribution of queue-lengths in a random switch-slot. Consider the queue-lengths as seen by departing packets. The queue-length Q_{n+1} as seen by the $(n+1)$st departing packet depends only on two quantities. The first is the queue-length Q_n as seen by the nth departing packet. The second is the state of the Markov chain modeling the arrivals in the switch-slot in which the nth packet departed. A switch-slot is called a *departure-switch-slot* if a packet departs in that switch-slot. The queue-length as seen by a departing packet together with the arrival process state in the departure-switch-slot form a 2-state DTMC over the state space

$$\{(i,j) : i \in \{0,1,\ldots\}, \quad j \in \{ON, OFF_1, OFF_2, \ldots, OFF_{r-1}\}.$$

The first state is the queue-length and the second state is the arrival process state. Let us rename the arrival process states as $ON = 0, OFF_1 = 1, OFF_2 = 2, \ldots, OFF_{r-1} = r-1$ for notational convenience. We then refer to the first state as the *level* and the second state as the *phase* of the DTMC.

As we are considering departure instants, the level (queue-length) can go down by at most 1 whereas it can go up by any amount in any transition of the DTMC depending on the number of arrivals. We club the states with the same level. As there are r phases for each level, there may be $(r \times r = r^2)$ possibilities for each level transition. In fact, the transitions from level $i+1$, $i \geqslant 0$, to level $k+i$ are governed by an $r \times r$ matrix which we denote by A_k^i, $k \geqslant 0$ whereas the transitions from the boundary level 0 to level k are governed by an $r \times r$ matrix which we denote by B_k, $k \geqslant 0$. Therefore, the transition matrix $\mathcal{P}$ of the 2-D DTMC takes the form,

$$\mathcal{P} = \begin{bmatrix} B_0 & B_1 & B_2 & B_3 & \cdots \\ A_0^0 & A_1^0 & A_2^0 & A_3^0 & \cdots \\ \tilde{0} & A_0^1 & A_1^1 & A_2^1 & \cdots \\ \tilde{0} & \tilde{0} & A_0^2 & A_1^2 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}, \tag{22}$$

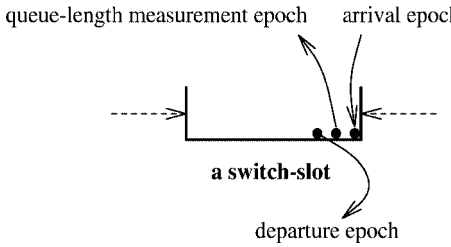


Fig. 5. Arrival, departure and measurement epochs for input queue analysis.

where $\tilde{0}$ is an $r \times r$ zero matrix. Further, consider two successive departure-switch-slots, say, nth and $(n+1)$st. Suppose, at the nth departure-switch-slot, the 2-D DTMC is in state (i, x) and at the $(n+1)$st departure-switch-slot, it is in state (j, y) and $i \geqslant 0$. The transition probability from state (i, x) to state (j, y) depends only on the difference $(j - i + 1)$ and the phases x and y, where $(j - i + 1)$ gives the number of arrivals between the two departures. This fact simplifies the $\mathcal{P}$ matrix further. We can then say that, the transitions from level $i+1,\ i \geqslant 0$ to level $i+k,\ k \geqslant 0$ are governed by the matrix $A_k,\ k \geqslant 0$, or

$$
\begin{aligned}
A_0 &= A_0^0 = A_0^1 = A_0^2 = \cdots = A_0^i = \cdots \\
A_1 &= A_1^0 = A_1^1 = A_1^2 = \cdots = A_1^i = \cdots \\
&\vdots \\
A_k &= A_k^0 = A_k^1 = A_k^2 = \cdots = A_k^i = \cdots \\
&\vdots
\end{aligned}
$$

The $\mathcal{P}$ matrix then takes the form,

$$
\mathcal{P} = \begin{bmatrix}
B_0 & B_1 & B_2 & B_3 & \cdots \\
A_0 & A_1 & A_2 & A_3 & \cdots \\
\tilde{0} & A_0 & A_1 & A_2 & \cdots \\
\tilde{0} & \tilde{0} & A_0 & A_1 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \ddots
\end{bmatrix}. \tag{23}
$$

We have deduced the structure of the $\mathcal{P}$ matrix without knowing what the transition probabilities actually are. We now proceed to compute the transition probabilities. The precise epochs of a packet arrival (if any), a departure (if any) and of queue-length measurement in any switch-slot are shown in Fig. 5. Also, a packet arriving in a switch-slot can start service only from the next switch-slot.

The transition probabilities are the elements of the matrices A_k and B_k. We derive the expression for one element of these matrices for illustration. The other elements can be derived in a similar fashion. Please refer to [8] for their expressions.

Consider the A_k matrix. The elements of A_k are $a_{i,j}^k$ where $0 \leqslant i \leqslant (r-1),\ 0 \leqslant j \leqslant (r-1)$. $a_{i,j}^k$ is the probability of transition from the state (l, i) in say, the nth departure-switch-slot, to the state $(l-1+k, j)$ in the $(n+1)$st departure-switch-slot where $l > 0$. The number of switch-slots elapsed between these two departure-switch-slots is the service time of the $(n+1)$st departing packet. The distribution of the service time, $\Pr\{D = d\}$, was calculated in Section 3.2.1. We write,

$$\Pr\{D = d\} = c_d, \quad d = 1, 2, 3, \ldots$$

If there are k arrivals during this service time and if the arrivals occur in a way so as to leave the arrival Markov chain in phase j at the end of the service time of this $(n+1)$st packet, only then the state of the 2-D DTMC changes to $(l-k+1, j)$. Assume that the $(n+1)$st departure received d switch-slots of service, say $\{1, 2, \ldots, d\}$ switch-slots. Given that the arrival Markov chain is in phase i in the 0th switch-slot, let $u^k_{i,j,d}$ be the probability that,

– there are k arrivals during the nth and $(n+1)$st departure epochs and
– the arrival Markov chain is in phase j in the dth switch-slot.

Therefore $a^k_{i,j}$ can be written as,

$$a^k_{i,j} = \sum_{d=1}^{\infty} u^k_{i,j,d} \cdot c_d. \tag{24}$$

It is not always possible to get closed form expressions for the $u^k_{i,j,d}$. We can get closed form expressions in this case due to the nice structure of the arrival Markov chain. Please refer to [8] for these expressions. We give a method in the Appendix 6 for a more general arrival Markov chain which computes the $u^k_{i,j,d}$. As the method is applicable to this arrival Markov chain, it can also be used instead of the closed form expressions for computing the $u^k_{i,j,d}$.

To get the steady-state probability vector of the $\mathcal{P}$ matrix, we need to solve the set of equations,

$$\tilde{\pi} = \tilde{\pi}\mathcal{P},$$
$$\|\tilde{\pi}\| = 1,$$

where $\tilde{\pi}$ is a vector of the form,

$$\tilde{\pi} = \left[\pi^0\ \pi^1\ \pi^2\ \cdots\ \pi^i\ \cdots\right],$$

and π^i is a vector of the form,

$$\pi^i = [\pi_{i,0}\ \pi_{i,1}\ \pi_{i,2}\ \cdots\ \pi_{i,r-1}].$$

$\pi_{i,j}$ is the steady-state probability of the state (i, j) of the 2-D DTMC.

Note that the $\mathcal{P}$ matrix is a structured $M/G/1$-type of matrix. We use TELPACK [1] to compute the steady-state probability vector $\tilde{\pi}$ of our $\mathcal{P}$ matrix. The steady-state probability vector $\tilde{\pi}$ enables us to compute the steady-state distribution of queue-length as seen by a departing packet. The steady-state queue-length vector is of the form,

$$\tilde{q} = [q_0\ q_1\ q_2\ \cdots\ q_i\ \cdots],$$

where,

$$q_i = \sum_{j=0}^{r-1} \pi_{i,j}. \tag{25}$$

The moments of the distribution of the queue-length as seen by a departing packet are then numerically computed using the above equations. The transition rates into and out of each state in a discrete-state stochastic process must be identical. By using the time-average interpretations for the steady-state queue-length probabilities, we see that the distribution of queue-length as seen by a departing packet is the same as the distribution of queue-length as seen

by an arriving packet [26, pp. 387–388]. Note that in [26, pp. 387–388], the Markov chain is one-dimensional. In our case it is two-dimensional. But as we are interested in the steady-state probability vector of levels (i.e., queue-lengths), the arguments are applicable in this case also. We compare the results of the above analysis with the results of simulations in Section 4. As it is not possible to compute the distribution of queue-length in a random slot, we cannot analyze the delay of a packet in the input queue. In Section 5, we show simulation results for the delay of a packet in the input queue.

3.2.3. Output queue analysis

We have assumed that, as $N \to \infty$, all contention queue processes are identical and independent. It then follows that all output queue systems are also identical and independent. It is sufficient to analyze only one output queue. It is difficult to characterize the departure process of a contention queue due to its general queueing model structure. The exact arrival process at the output queue is then unknown. We use an approximate ON-OFF type of arrival model to analyze the output queue. An exact analysis like that of input queues may be difficult in this case. A packet arrives in the output queue after s switch-slots in a busy period. This is because the contention queue service time is s switch-slots. Therefore we model the arrival process as follows:

A packet arrives at output queue i in a switch-slot only if the switch status is T_{s-1} at the beginning of the switch-slot (Section 3.2.1.2). There are always $(s-1)$ idle switch-slots (corresponding to the switch status T_0 to T_{s-2}) before an arrival in the output queue. The group of all these switch-slots is called a *busy cluster*. In the busy period of the contention queue, packets depart regularly at the end of the busy clusters. When the contention queue is empty in a switch-slot, no packet arrives in output queue i in that switch-slot. We construct the ON-OFF arrival model such that the mean busy period of the contention queue is equal to s times the mean number of busy clusters. The mean OFF period is equal to the mean idle period of contention queue i.

We turn our attention to contention queue i again. It is necessary to compute the distribution of the idle period I of this queue. Note that $\Pr\{I \geqslant 1\} = 1$ because the idle period is at least one switch-slot. The idle period ends when there is an arrival in the contention queue i and arrivals to the contention queue i are Batch-Poisson with rate λ given by (1). In other words, the idle period continues with probability $e^{-\lambda}$ (probability of no packet arrivals in a switch-slot) and ends with probability $1 - e^{-\lambda}$ (probability of at least one arrival in a switch-slot). It follows that,

$$\Pr\{I = 2\} = e^{-\lambda} \cdot \left(1 - e^{-\lambda}\right), \tag{26}$$

$$\Pr\{I = 3\} = e^{-\lambda} \cdot e^{-\lambda} \cdot \left(1 - e^{-\lambda}\right) \tag{27}$$

$$\vdots$$

$$\Pr\{I = i\} = e^{-(i-1)\lambda} \cdot \left(1 - e^{-\lambda}\right) \tag{28}$$

$$\vdots$$

Using (26)–(28) the mean idle period which is,

$$E[I] = \sum_{i=1}^{\infty} i \cdot \Pr\{I = i\},$$

can be computed. It turns out that,

$$E[I] = \frac{1}{1 - e^{-\lambda}}. \tag{29}$$

The utilization ρ of the contention queue is

$$\begin{aligned} \rho &= \text{arrival rate} \times \text{average service time} \\ &= \lambda \times s \\ &= s\lambda, \end{aligned} \tag{30}$$

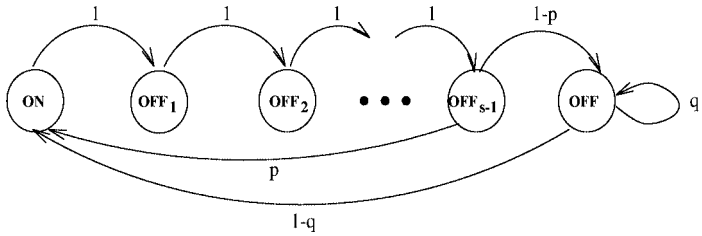


Fig. 6. Approximate model of the arrival process in the output queues for a rational speed-up factor r/s.

because the contention queue is modeled as a single server system. The utilization ρ is also the fraction of the time the server is busy, or,

$$\rho = \frac{E[B]}{E[I] + E[B]}. \tag{31}$$

Using (29), (30) and (31) we get,

$$E[B] = \frac{s\lambda}{(1 - s\lambda)(1 - e^{-\lambda})}. \tag{32}$$

Note again that, when the contention queue is busy serving a packet, there are no departures from the contention queue. The busy period is always an integer multiple of s which is the fixed service time. So there are always at least $(s-1)$ idle switch-slots between two successive departures. Also a packet departed from the contention queue enters the output queue. The ON-OFF Markov process takes this fact into account. The ON-OFF Markov process is depicted in Fig. 6. The ON and OFF_i form a busy cluster. The p and q are computed as follows,

$$E[\text{Busy Clusters}] = 1/(1 - p),$$
$$s \cdot E[\text{Busy Clusters}] = E[B].$$

Therefore,

$$p = 1 - s/E[B], \tag{33}$$
$$E[\text{OFF}] = 1/(1 - q),$$
$$E[\text{OFF}] = E[I].$$

Therefore,

$$q = 1 - 1/E[I]. \tag{34}$$

Note that for *integer* speed-up factors $s = 1$. There are no idle slots in a busy cluster and a busy cluster consists of only ON slot. The ON-OFF Markov chain of Fig. 6 reduces to the simple ON-OFF Markov chain of Fig. 7.

The service time of the output queue is r switch-slots and is deterministic (the transmission time of a packet on the output link). We analyze this output queue model. The queue-length in a random switch-slot cannot be modeled by a DTMC. We look at the departure-switch-slots. The output queue-length as seen by a departing packet in a switch-slot together with the ON-OFF chain status in that switch-slot form a 2-D DTMC over the state space $\{(i, j) : i \geqslant 0, j \in \{ON, OFF_1, OFF_2, \ldots, OFF_{s-1}, OFF\}\}$ where i is the queue-length in a departure-switch-slot and j is the state of the ON-OFF process. Let $ON \equiv s$ and $OFF \equiv 0$ and $OFF_j \equiv j$. We say that i is the

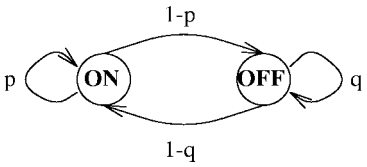


Fig. 7. Approximate model of the arrival process in the output queues for an integer speed-up factor.

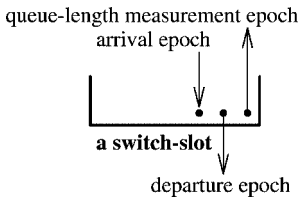


Fig. 8. Arrival, departure and measurement epochs for the analysis of output queues.

level of the 2-D DTMC and j is the phase of the 2-D DTMC. The precise epochs of packet arrival (if any), packet departure (if any) and queue-length measurement in a switch-slot are shown in Fig. 8. Note that in this case the level i can go down by at most 1 in a transition of the 2-D DTMC and it can go up by at most $m = \lceil r/s \rceil$ in a transition of the 2-D DTMC. We club all the states having the same level so that a level transition is represented by a $(s+1) \times (s+1)$ matrix. Matrix B_k represents a transition from level 0 to level k. Matrix C represents transition from level 1 to level 0 and matrix A_k represents transition from a level i to a level $j = i + k - 1$ where $i > 0$ by arguing as in the case of input queue model. The state transition matrix is, then, of the form,

$$\mathcal{P} = \begin{bmatrix} B_0 & B_1 & B_2 & \cdots & B_m & \tilde{0} & \tilde{0} & \tilde{0} & \cdots \\ C & A_1 & A_2 & \cdots & A_m & \tilde{0} & \tilde{0} & \tilde{0} & \cdots \\ \tilde{0} & A_0 & A_1 & \cdots & A_{m-1} & A_m & \tilde{0} & \tilde{0} & \cdots \\ \tilde{0} & \tilde{0} & A_0 & \cdots & A_{m-2} & A_{m-1} & A_m & \tilde{0} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}. \tag{35}$$

An element $b_{i,j}^k$ of B_k is given by,

$$b_{i,j}^k = \Pr\{\text{there is a transition from the state } (0, i) \text{ to the state } (k, j)\}.$$

An element $c_{i,j}$ of C_k is given by,

$$c_{i,j} = \Pr\{\text{there is a transition from the state } (1, i) \text{ to the state } (0, j)\}.$$

An element $a_{i,j}^k$ of A_k is given by,

$$a_{i,j}^k = \Pr\{\text{there is a transition from the state } (x, i) \text{ to the state } (x + k - 1, j)\},$$

where $x > 0$, and x is the row number where the A_k matrix appears in the $\mathcal{P}$ matrix. By using the method in Appendix A, we can compute all the sub-matrices and consequently, the whole $\mathcal{P}$ matrix. Note that the $\mathcal{P}$ matrix is a structured $M/G/1$-type matrix. The steady-state probability vector of the $\mathcal{P}$ matrix is computed by using TELPACK. The steady-state probability vector is of the form,

$$\pi = \left[\pi^0 \ \pi^1 \ \pi^2 \ \cdots \ \pi^i \ \cdots\right],$$

where,

$$\pi^i = \left[\pi_{i,0}\ \pi_{i,1}\ \cdots\ \pi_{i,s}\right].$$

The steady-state probability vector of queue-lengths as seen by a departing packet is of the form,

$$\tilde{q} = [q_0\ q_1\ q_2\ \cdots\ q_i\ \cdots],$$

where,

$$q_i = \pi_{i,0} + \pi_{i,1} + \cdots + \pi_{i,s}.$$

The results of the above analysis are compared with the results of simulations in the next section of this chapter. It is not possible to compute the distribution of the queue-length in a random switch-slot due to the model of the output queue. Therefore, we are unable to analyze the delay performance of this output queue. In Section 5 of this chapter, we show the results of simulations for the delay of a packet in the output queue.

4. Analytical vs simulated performance

Simulations of the actual switch-slotted speeded-up switch fabrics were performed to test the validity of various assumptions made in the analysis and to substantiate the asymptotic analysis results. We have considered a 64×64 nonblocking speeded-up switch fabric. Results for three speed-up factors are studied: $1.5(= 3/2)$, $2(= 2/1)$ and $3(= 3/1)$. The analysis of the input and output queues gives the mean queue-length as seen by a departing packet. We obtain the same data from simuation for comparison. We plot the mean input queue-length obtained by analysis and simulation in Fig. 9. It can be seen that the results are in good agreement. Note that even though the input queue analysis is exact, it is asymptotic, and thus we compare it with simulations to judge the effect of finite switch sizes. The mean output queue-length as seen by a departure is plotted in Fig. 10. It can be seen that there is a slight discrepancy between the analytical and simulated results. The reason for this discrepancy is the approximation of the arrival process to the output queue by the ON-OFF process. Observe that the discrepancy is more pronounced for the speed-up factor 1.5. For speed-ups of 2 and 3 the results of both cases are in good agreement. Thus the approximation works better for higher speed-up factors. Overall, the asymptotic analysis results conform well with the simulation results.

5. More simulation results

5.1. Delay for modified Bernoulli traffic

We study the simualtion results of the average delay of the speeded-up packet switches for the arrival process of Section 3.1. We plot the total average delay of a packet in the switch obtained through simulations in Fig. 11. The average delays for the cases of pure input-queued and pure-output queued switches are also plotted in Fig. 11. It can be seen from the plot that the delay curves for speed-up factors 2 and 3 are close to the delay curve of the pure output-queued switch. From the delay curves it can be seen that a speedup factor between 1.5 and 2 is enough to approximately achieve pure output-queued switch performance. This is an important observation considering the fact that the performance of pure output-queued switches is the best. The interesting result is that, even with a speed-up of 2, the performance of the pure output-queued switch (speed-up factor N) can be closely approximated. In the case of a pure input-queued switch, the HOL blocking phenomenon plays a major role in degrading the performance. Even a moderate speed-up of 2 quickly overcomes the HOL blocking phenomenon.

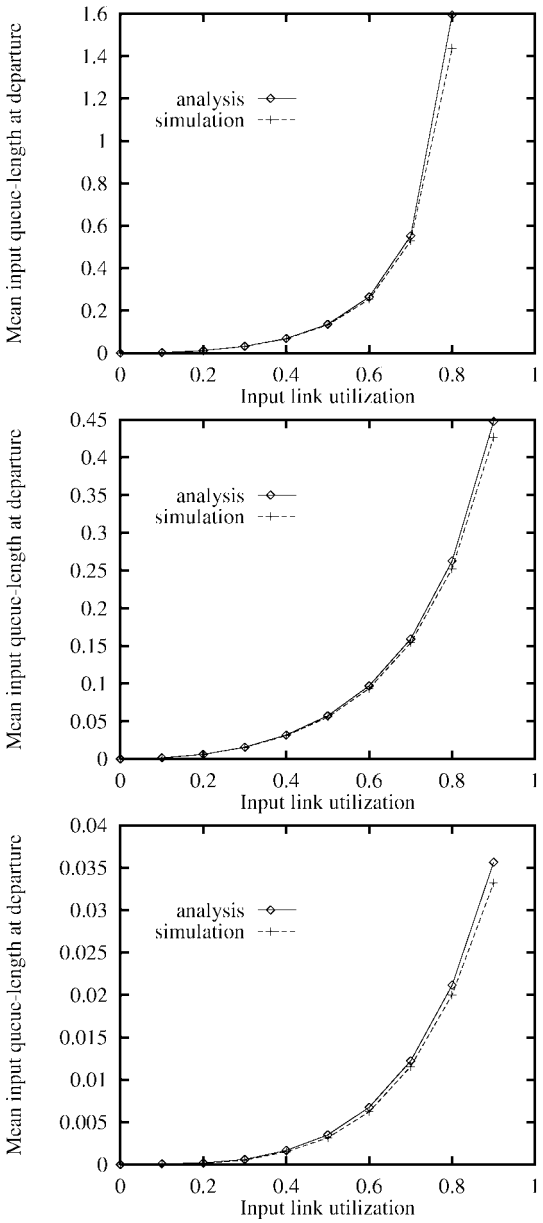


Fig. 9. The mean input queue-length as seen by a departure obtained by analysis is compared to that obtained by simulation, (top) speed-up of 1.5, (middle) speed-up of 2, (bottom) speed-up of 3. The input link utilization is $r\lambda$, see (1). The arrival process is of Section 3.1.

We have plotted the input and output queue delays separately for a speed-up factor of 2 in Fig. 12. It can be seen that the total delay is dominated by the output queueing delay. This result is practically useful for the design and operation of high speed switch fabrics, more so with ever-increasing link speeds. The speeded-up switch fabric (with a modest speed-up of 2) has largely overcome the HOL blocking problem of pure input-queued switches and the high switch speed (precisely N times) problem of pure output-queued switches and closely approximates the best performance obtainable.

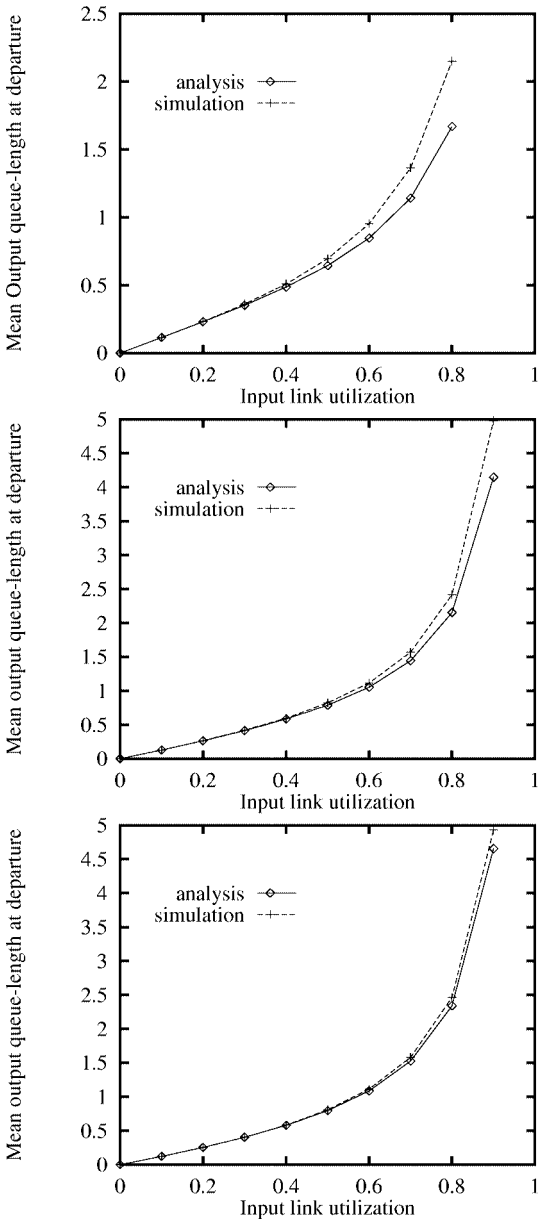


Fig. 10. The mean output queue-length as seen by a departure obtained by analysis is compared to that obtained by simulation, (top) speed-up of 1.5, (middle) speed-up of 2, (bottom) speed-up of 3. The input link utilization is $r\lambda$, see (1). The arrival process is of Section 3.1.

5.2. Delay for bursty traffic

We now consider bursty traffic with the following model. 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. 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 r. v. with mean $b - 1$. For a mean burst size $b = 50$, the saturation throughput calculated from the result in [18] is 50.6%, for large N. For a finite switch size, 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

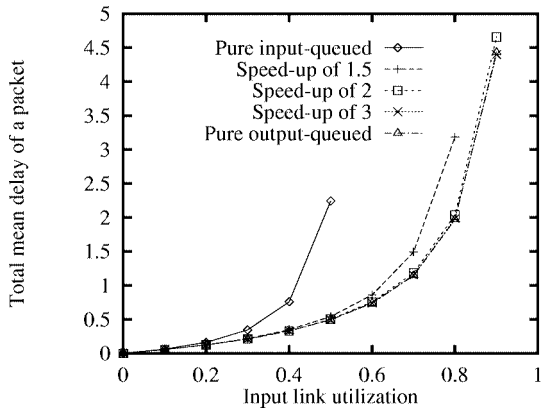


Fig. 11. The mean total packet delay results for switches with speed-up factors of 1.5, 2 and 3 are compared with the delay results of pure input-queued and pure output-queued switches. All the results are obtained by simulations. The input link utilization is $r\lambda$, see (1). The arrival process is of Section 3.1.

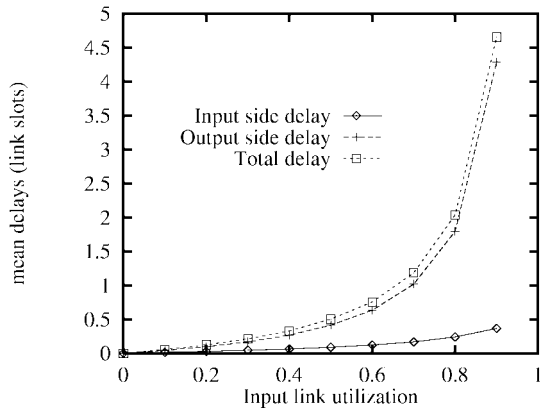


Fig. 12. The mean delay in input queue and the mean delay in output queue are compared to the total mean delay of a packet for a speed-up factor of 2. All the results are obtained by simulations. The input link utilization is $r\lambda$, see (1). The arrival process is of Section 3.1.

full link utilization. The mean queueing delays achieved by output queueing and input queueing with a speedup factor of 2 are shown in Fig. 13. 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, Fig. 14 shows the mean queueing delays for an average burst size $b = 50$.

5.3. Switch fabrics with memory

Another option to overcome the throughput limitations of pure input-queued switches, that has been investigated, is to use switch fabrics with memory [12]. 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

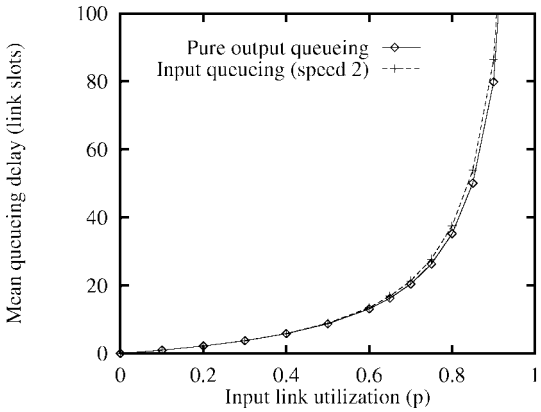


Fig. 13. The delay performance of a switch with speed-up of 2 is compared with the delay performance of pure output-queued switch. All the results are obtained by simulations. The arrival of packets on each input link is bursty with a mean burst size $b = 10$. The expected number of packets in each link-slot equals p.

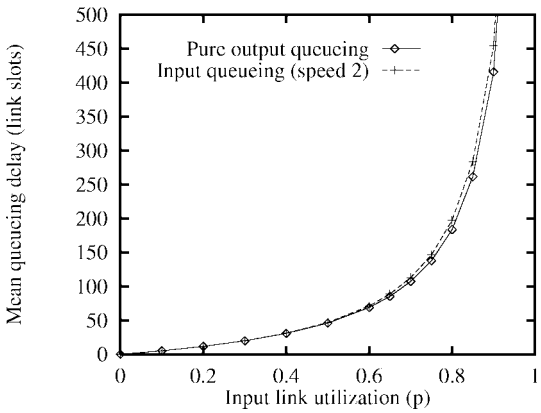


Fig. 14. The delay performance of a switch with speed-up of 2 is compared with the delay performance of pure output-queued switch. All the results are obtained using simulations. The arrival of packets on each input link is bursty with a mean burst size $b = 50$. The expected number of packets in each link-slot equals p.

pure output-queued switches when m is large. Illiadis and Denzel [12] 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 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 link utilization. For example, for $m = 8$, a speedup factor of $1.2 > 1/0.8426$ is sufficient. In Fig. 15, we have plotted the mean queueing delays for a switch fabric with $m = 8$ without speedup and with a speedup factor of 1.2. The number of switch ports is $N = 16$ and the packet destinations are chosen independently ($b = 1$). The perfrmance of pure 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 pure output queueing, when the switch fabric is operated at a speedup factor of 1.2.

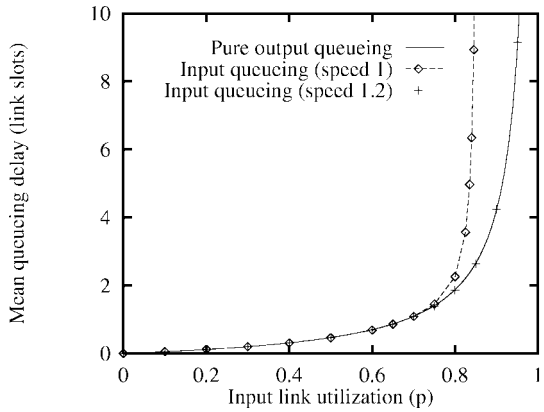


Fig. 15. The delay performance obtained by simulations of a switch fabric with a memory within the fabric of eight packets per output port, with a speedup factor of 1 (no speedup) and a speedup factor of 1.2, is compared with the delay performance of pure output queueing. The mean burst size $b = 1$ so that the destinations of successive packets are independent. The expected number of packets in each link slot equals p.

Using memory within the switch fabric can be expensive, especially if it is to be implemented using on-chip memory; see, for example, [7]. Therefore, reducing the required memory size by speeding up the fabric, even if 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 [7] 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 memoy 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 2. This phenomenon is illustrated in Fig. 16 where we compare the mean queueing delays achieved without speedup and with a speedup factor of 2, with the mean queueing delay achieved by pure output queueing for bursty traffic with a mean burst size $b = 50$ cells.

6. Conclusions

In this paper, we have used matrix-geometric techniques to investigate the impact of speedup on the performance of combined input and output queueing packet switches. We have also used simulations to investigate the effect of bursty traffic and the provision of memory within the switch fabric. The important observation is that only a moderate speedup factor (about two) is necessary to approach not only the throughput but also the delay performance of pure output-queued switches. Moreover, these results hold for random scheduling of the head-of line packets; no complex scheduling and/or matching algorithms are required within the switch. This is practically significant for the design and operation of high speed switch fabrics, especially with the continued increase in link speeds and switch sizes.

Appendix: Probabilities of packet arrivals in a service time

In this appendix we present a programmable method to calculate the probabilities of the numbers of arrivals in a service time when the initial and final state of the arrival chain are known. We have already seen in Section 3.2.2

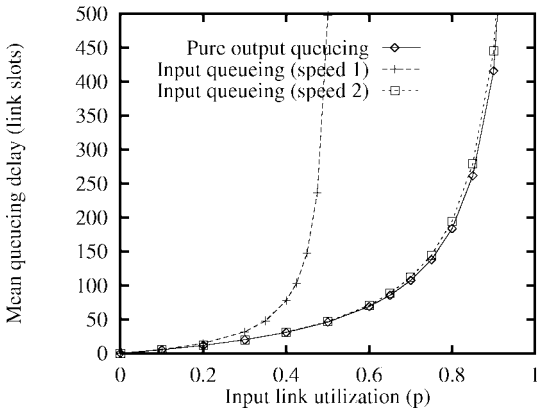


Fig. 16. The delay performance obtained by simulations of a switch fabric with a memory within the fabric of eight packets per output port, with a speedup factor of 1 (no speedup) and a speedup factor of 2, is compared with the delay performance of pure output queueing. All the results are obtained using 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.

that these probabilities are required to compute the complete state transition matrix of the queue-length as seen by a departing packet. In that section we had a nicely structured arrival process which made it possible to obtain closed form expressions. This is not always the case. We use the method described here to compute complete state transition matrices in Section 3.2.2. In fact, this method can be used for any arrival process which is modeled by a Markov chain. It may have more general applications as well.

To illustrate the method, consider an ON-OFF arrival process which has two ON states, ON_1 and ON_2 and one OFF state. Any of the ON states signifies an arrival in a slot and an OFF state signifies an idle slot. The transition matrix is of the form,

$$\begin{bmatrix} q & (1-q) & 0 \\ (1-p) & 0 & p \\ 1 & 0 & 0 \end{bmatrix}, \tag{36}$$

where, $0 < p < 1$ and $0 < q < 1$ and the 1st row gives all one-step transitions from the OFF state, the 2nd row gives all one-step transitions from the ON_1 state and the 3rd row gives all one-step transitions from the ON_2 state.

Supposing that the service time is d slots and the initial and final states of the arrival chain are known, we want to compute the probabilities of the numbers of packet arrivals in these d slots. We use a variable x to keep track of the number of arrivals and it's associated probability as follows. A packet arrives only when there is a transition to one of the ON states. We multiply such transition probabilities in the above matrix by x. The new matrix has the form,

$$\begin{bmatrix} q & (1-q)x & 0 \\ (1-p) & 0 & px \\ 1 & 0 & 0 \end{bmatrix}. \tag{37}$$

Note that x is just a variable and no value (quantity) is associated with it. Now consider the case when $d = 2$. That is, we want to compute the probability of i arrivals in 2 consecutive slots for all combinations of initial and final states. For this we square the above matrix and we get a new matrix which is as follows,

$$\begin{bmatrix} q^2 + (1-q)(1-p)x & q(1-q)x & (1-q)px^2 \\ q(1-p) + px & (1-p)(1-q)x & 0 \\ q & (1-q)x & 0 \end{bmatrix}. \tag{38}$$

Now consider coefficients of various powers of x in the matrix elements. Say, the transitions occurred over the slots $\{0, 1, 2\}$. Observe the (1, 1) element of the above matrix which is $q^2 + (1 - q)(1 - p)x$. In this the coefficient of x^0 gives the probability that there are 0 arrivals over slots 1 and 2 (i.e., service time $d = 2$) when the arrival chain was in the OFF state in the 0th slot and it is in the OFF state again in the 2nd slot. Thus the coefficient of x^i in the expression for the entry (j, k) of the matrix gives the probability that there are i arrivals in $d = 2$ service slots when the initial state of the arrival chain is j and the final state is k.

When $d = x$, we have to calculate the xth power of the matrix. By separating the coefficients of x^i from the expressions for the entries of the new matrix, we can get the required probabilities. Note that what we have calculated here is the $u^k_{i,j,d}$ of Section 3.2.2.

The advantage of this method is that it is easily programmable. In our work we have found *Maple* as a suitable software for this method. We have used *Maple* to program this method. Note that this method can be suitably extended to directly compute the required state transition matrix. It may also have applications in other contexts.

References

[1] A. Agrawal and N. Oguz, Telpack software, http://www.cstp.umkc.edu/org/tn/telpack/home.html

[2] N. Akar and K. Sohraby, An invariant subspace approach in M/G/1 and G/M/1 type Markov chains, Technical report, Computer Science Telecommunications, University of Missouri-Kansas City, MO, 64110, 1995.

[3] N. Akar and K. Sohraby, On computational aspects of the invariant subspace approach to teletraffic problems and comparisons, Technical report, Computer Science Telecommunications, University of Missouri-Kansas City, MO, 64110, 1995.

[4] D. Bini and B. Meini, On the solution of a nonlinear matrix equation arising in queueing problems, *SIAM Journal of Matrix Anal. Appl.* **17**(4) (1996), 906–926.

[5] A. Charny, P. Krishna, N. Patel and R. Simcoe, Algorithms for providing bandwidth and delay guarantees in input-buffered crossbars with speedup, in: *Proceedings of the Sixth International Workshop on Quality of Service (IWQoS '98)*, 1998.

[6] 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* **9**(3) (1991), 439–449.

[7] W.E. Denzel, A.P.J. Engbersen and I. Illiadis, A flexible shared buffer switch for ATM at Gb/s rates, *Computer Networks and ISDN Systems* **27**(4) (1995), 611–624.

[8] A.S. Diwan, Performance analysis of speeded-up high-speed packet switches, MSc (Engg) thesis, Indian Institute of Science, Bangalore, India, Feb., 1998.

[9] W. Feller, *An Introduction to Probability Theory and Its Applications*, Volume 1, third edition, Wiley Eastern Limited, 1991.

[10] R. Guérin and K. Sivarajan, Delay and throughput performance of speeded-up input-queueing packet switches, *IBM Research Report*, 1997.

[11] J.Y. Hui and E. Arthurs, A broadband packet switch for integrated transport, *IEEE Journal of Selected Areas in Communications* **5**(8) (1987), 1264–1273.

[12] I. Illiadis and W.E. Denzel, Analysis of packet switches with input and output queueing, *IEEE Transactions on Communications* **41**(5) (1993), 731–740.

[13] L. Jacob, Performance analysis of scheduling strategies for switching and multiplexing of multiclass variable bit rate traffic in an ATM network, Ph.D. thesis, Indian Institute of Science, Bangalore, India, Dec., 1992.

[14] L. Jacob and A. Kumar, Saturation throughput analysis of an input queueing atm switch with multiclass bursty traffic, *IEEE Transactions on Communications* **43**(2/3/4) (1995), 757–761.

[15] 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* **4**(2) (1996), 258–271.

[16] M. Karol, M. Hluchyj and S. Morgan, Input versus output queueing on a space division switch, *IEEE Transactions on Communications* **35**(12) (1987), 1347–1356.

[17] P. Krishna, N.S. Patel, A. Charny and R. Simcoe, On the speedup required for work-conserving crossbar switches, in: *Proceedings of the Sixth International Workshop on Quality of Service (IWQoS '98)*, 1998.

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

[19] S.C. Liew, Performance of various input-buffered and output-buffered atm switch design principles under bursty traffic: simulation study, *IEEE Transactions on Communications* **42**(2/3/4) (1994), 1371–1379.

[20] N. McKeown, V. Anantharam and J. Walrand, Achieving 100% throughput in an input-queued switch, in: *IEEE INFOCOM*, San Francisco, CA, 1996, pp. 296–302.

[21] M.F. Neuts, *Matrix-geometric Solutions in Stochastic Models*, John Hopkins University Press, Baltimore, MD, 1981.

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

[23] B. Prabhakar and N. McKeown, On the speedup required for combined input and output queued switching, Computer Systems Lab, Technical Report CSL-TR-97-738, Stanford University, 1997.

[24] V. Ramaswami, Tutorial on matrix analytic methods in queueing theory, in: *ATM Workshop*, 1995.

[25] I. Stoica and H. Zhang, Exact emulation of an output queueing switch by a combined input output queueing switch, in: *Proceedings of the Sixth International Workshop on Quality of Service (IWQoS '98)*, 1998.

[26] R.W. Wolff, *Stochastic Modeling and the Theory of Queues*, Prentice Hall, Englewood Cliffs, New Jersey, 1989.

[27] 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* **5**(8) (1987), 1274–1283.