Logarithmic Delay for $N \times N$ Packet Switches

Michael J. Neely
http://www-rcf.usc.edu/ mjneely
mjneely@usc.edu

Eytan Modiano
http://web.mit.edu/modiano/www
modiano@mit.edu

***Abstract*— We consider the fundamental delay bounds for scheduling packets in an $N \times N$ packet switch operating under the crossbar constraint. Algorithms that make scheduling decisions without considering queue backlog are shown to incur an average delay of at least $O(N)$. We then prove that $O(\log(N))$ delay is achievable with a simple frame based algorithm that uses queue backlog information. This is the best known delay bound for packet switches, and is the first analytical proof that sublinear delay is achievable in a packet switch with random inputs. The algorithm is shown to be implementable with very low complexity, requiring $O(N^{1.5}\log(N))$ total operations per timeslot.**

***Index Terms*—stochastic queueing analysis, scheduling, optimal control**

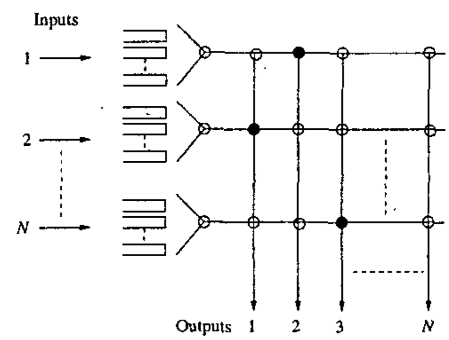

Fig. 1. An $N \times N$ packet switch under the crossbar constraint.

I. Introduction

We consider an $N \times N$ packet switch with N input ports an N output ports, shown in Fig. 1. The system operates in slotted time, and every timeslot packets randomly arrive at the inputs to be switched to their destinations. Scheduling is constrained so that each input can transfer at most one packet per timeslot, and outputs can receive at most one packet per timeslot. This constraint arises from the physical limitations of the *crossbar switch fabric* that is commonly used to transfer packets from inputs to outputs, and gives rise to a very rich problem in combinatorics and scheduling theory. This problem has been extensively studied over the past decade [1-19], and remains an important topic of current research. This is due to both its technological relevance to high speed switching systems and its pedagogical example as a network complex enough to inspire interesting research yet simple enough for an extensive network theory to be developed.

In this paper, we show that if the matrix of input rates to the switch has a sufficient number of non-negligible entries (to be made precise in Section III), then any scheduling strategy which does not consider queue backlog information necessarily incurs an average delay of at least $O(N)$. Strategies that do not consider backlog have been proposed in a variety of contexts, including work by Chang et al [2], [3], Leonardi et al [4], Koksal [5], and Andrews and Vojnović [6]. The basic idea is to construct a randomized or periodic scheduling rule precisely matched for known input rates. If these rates are indeed known a-priori and do not change with time, then such scheduling offers arbitrarily low per-timeslot computation complexity, as any startup complexity associated with computing the scheduling rule is mitigated as the same rule is repeatedly used for all time.

The $O(N)$ delay result introduces an intuitive tradeoff between delay and implementation complexity, as algorithms which do not consider backlog information may have lower complexity yet necessarily incur delay that grows linearly in the size of the switch. To improve delay, we construct a simple algorithm called Fair-Frame that uses queue backlog information when making scheduling decisions. For independent Poisson inputs, we show that the Fair-Frame algorithm stabilizes the system and provides $O(\log(N))$ delay whenever the input rates are within the switch capacity region. This work for the first time establishes that sub-linear delay is possible in an $N \times N$ switch. Furthermore, the proof is simple and provides the intuition that logarithmic delay is achievable in any single-hop network with a capacity region that is described by a polynomial number of constraints. Such delay improvement is achieved by taking advantage of statistical multiplexing gains, which is not possible for backlog-unaware algorithms.

Previous work in scheduling is found in [1-19]. In [2] it is shown that stable scheduling can be achieved with a queue length-oblivious strategy by using a Birkhoff-Von Neumann decomposition on the known arrival rate matrix. In [7] and [8], it was shown that scheduling according to an $O(N^3)$ *Maximum Weighted Match* (MWM) every timeslot stabilizes the switch whenever possible without requiring prior knowledge of the input rates. In [9] the delay of the MWM algorithm was shown to be no more than $O(N)$. We note that MWM scheduling is queue length-aware, and hence it may be possible to tighten the delay bound to less than $O(N)$, as is suggested in the simulations of [9]. However, $O(N)$ delay is the tightest known analytical bound for MWM scheduling, and was previously the best known delay bound for *any* algorithm for a switch with random (Poisson) inputs.

In [10] it is shown that if a switch has an internal *speedup of* 2 (allowing for two packet transfers from input to output every timeslot), then exact output queue emulation can be achieved via stable marriage matchings, yielding optimal $O(1)$ delay. To date, there are no known delay optimal scheduling strategies for packet switches without speedup. However, in the landmark paper [11], a loss-rate optimal scheduling algorithm is constructed for a 2×2 switch with finite buffers. Finite buffer analysis of loss rates for systems of parallel queues is addressed in [12] [13].

Frame based approaches for stabilizing switches and networks with deterministically constrained traffic are considered in [14] [15] [16], and in [17] it was shown that a frame based algorithm using "greedy" maximum size matches can be used to stabilize an $N \times N$ packet switch with Poisson inputs. Complexity and delay tradeoffs are explored in [18], where an explicit complexity-delay curve is established allowing for stable scheduling at any arbitrarily low computation complexity with a corresponding tradeoff in average delay. Similar complexity reductions were developed in [19]. In this paper, we show that the $(complexity, delay)$ operating point of the Fair-Frame algorithm sits below the curve achieved by the class of algorithms given in [18]. Indeed, Fair-Frame offers logarithmic delay and can be implemented with $O(N^{1.5}\log(N))$ total operations per timeslot. The combination of low complexity and low delay makes Fair-Frame competitive even with output queue emulation strategies in switches with a speedup of 2.

II. Packet Switches and the Crossbar Constraint

Consider the $N \times N$ packet switch in Fig. 1. At each input, memory is sectioned into distinct storage buffers to form N *virtual input queues*, one for each destination. Packets randomly arrive to each input every timeslot and are placed in the virtual input queues according to their destinations. Note that there are a total of N^2 virtual input queues, indexed by (i, j) for $i, j \in \{1, \ldots, N\}$, where queue (i, j) holds data at input i destined for output j.

Packets arrive to the queues every timeslot according to arrival processes $A_{ij}(t)$. A scheduler selects a group of packets to switch from inputs to outputs by connecting the crosspoints in Fig. 1 according to a *permutation matrix* $(S_{ij}(t))$, so that no input or output port is scheduled for more than one packet transfer (where the matrix $(S_{ij}(t))$ is a $0-1$ matrix with exactly one '1' in each row and column, corresponding to the chosen crosspoint connections). The goal of the scheduler is to choose permutations every timeslot so that the overall system is stabilized and packets have bounded average queueing delay.

A. Stability and Delay

Assume inputs $A_{ij}(t)$ are rate ergodic and define the rate to queue (i, j) as $\lambda_{ij} \triangleq \lim_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t} A_{ij}(\tau)$. The *capacity region* of the switch is defined as the closure of the set of all rate matrices (λ_{ij}) which can be stabilized by the switch by using some scheduling algorithm, and is described by the following constraints:

$$\sum_{j=1}^{N} \lambda_{ij} \leq 1 \text{ for all inputs } i \tag{1}$$

$$\sum_{i=1}^{N} \lambda_{ij} \leq 1 \text{ for all outputs } j \tag{2}$$

If one of the above inequalities is violated, some input or output port must be overloaded, leading to instability. As an example of a stabilizing algorithm for the case when the rate matrix satisfies the above constraints, suppose traffic is Poisson with uniform rates, so that $\lambda_{ij} = \rho/N$ for all (i, j) (where $0 < \rho < 1$). Randomly choosing a permutation among all possible permutation matrices turns each virtual queue (i, j) into a discrete $M/GI/1$ queue with geometric service times with service probability $1/N$. Such a queue is clearly stable, and average delay can easily be easily calculated (see [20]):

$$\overline{W}_{\text{randomized}} = \frac{N-1/2}{1-\rho} + 1 \tag{3}$$

One can also consider periodic scheduling, where on slot t each input port i is connected to the output port $(i+t) \mod N$. The resulting queues are similar to $M/D/1$ queues, and delay is given by:

$$\overline{W}_{\text{periodic schedule}} = \frac{N}{2(1-\rho)} + 1 \tag{4}$$

While periodic scheduling reduces delay by a factor of 2, the delay remains $O(N)$. Intuitively, this is because each input port can service at most one of its N queues per timeslot, and hence it takes an average of $N/2$ timeslots for an arriving packet to see a server. In the next section we elaborate on this intuition to show that $O(N)$ delay is incurred by any scheduling algorithm that operates independently of the input streams and current levels of queue backlog.

We note that an output-queued system, which bypasses all switching and sends inputs directly to their output ports, has a delay given by:

$$\overline{W}_{\text{output queue}} = \frac{1}{2(1-\rho)} + 1 \tag{5}$$

where we again assume traffic is uniform and Poisson. This gap between $O(1)$ delay and $O(N)$ delay motivates our search for sublinear delay algorithms among the class of backlog-aware strategies, considered in Section IV.

III. An O(N) Delay Bound for Backlog-Independent Scheduling

Consider an $N \times N$ packet switch with general stochastic inputs arriving to each of the N^2 input queues. All inputs are assumed to be stationary and ergodic. Assume the system is initially empty and let $X_{ij}(t)$ represent the arrivals to input queue (i, j) during the interval $[0, t]$ (i.e., $X_{ij}(t) = \sum_{\tau=0}^{t} A_{ij}(\tau)$). Let $L_{ij}(t)$ represent the current number of packets queued at input (i, j), and let $S_{ij}(t)$ represent the server control decision at slot t (where the matrix $(S_{ij}(t))$ is a permutation matrix). Here we show that if the control decisions $(S_{ij}(t))$ are stationary and independent of the arrival streams, then average delay in the switch is necessarily $O(N)$.[1] Because backlog is directly related to the arrival streams, it follows that stationary switching schemes which operate independently of queue backlog incur at least $O(N)$ delay.

As a caveat, we note that periodic scheduling streams such as those proposed for Birkhoff-Von Neumann scheduling in [2] are by definition not stationary. However, randomizing the periodic schedule $(S_{ij}(t))$ over the phase of the period yields a

[1] The $O(N)$ result holds only when the input rate matrix (λ_{ij}) has a sufficient number of positive entries, as described in Theorem 2.

stationary schedule. If inputs are ergodic, stationary, and independent of the scheduling decisions, then the resulting average packet delay is the same under both the original periodic schedule and the schedule with a randomized phase. Thus, the $O(N)$ delay result also holds for any scheduling algorithm which is independent of backlog and which can be made stationary by phase randomization.

The following lemma is useful for obtaining lower bounds on delay. The proof uses a technique similar to that used in [21] to show that fixed length packets minimize delay over all packet length distributions with the same mean, and is given in [1].

Lemma 1. *For a switch with general arrival processes, any stationary scheduling algorithm which operates independently of the input streams (and hence, independently of the current queue backlog) yields a time average queue occupancy $\overline{L}_{ij}$ for each queue (i, j) satisfying:*

$$\overline{L}_{ij} \geq \overline{U}_{ij}$$

where U_{ij} represents the unfinished work *(or "fractional packets") in a system with the same inputs but with constant server rates of μ_{ij} packets/slot, for at least one set of rates μ_{ij} such that $\sum_j \mu_{ij} \leq 1$ for all i, and $\sum_i \mu_{ij} \leq 1$ for all j.*

Proof. The lemma is proved in [1]. □

The lemma above produces a lower bound on delay in terms of a system of queues with the same inputs but with constant server rates, and leads to the following theorem.

Theorem 1. *If inputs X_{ij} are Poisson with uniform rates $\lambda_{ij} = \rho/N$ (for $\rho < 1$ representing the loading on each input), then delay under any stationary scheduling algorithm which does not consider backlog is greater than or equal to $\frac{N}{2(1-\rho)}$.*

Proof. The unfinished work in an M/D/1 queue with arrival rate λ_{ij} and service time $1/\mu_{ij}$ is equal to $\overline{U}_{ij}(\mu_{ij}) = \frac{\lambda_{ij}}{2(\mu_{ij}-\lambda_{ij})}$, which can be computed by adding $\rho_{ij}/2$, the average portion of a packet remaining in the server, to the expression for the average number of packets in the buffer of an M/D/1 queue [20]. From Lemma 1, there exists a rate matrix (μ_{ij}) with row and column sums bounded by 1, so that $\overline{L}_{ij} \geq \overline{U}_{ij}$ for all (i, j). Define Λ as the set of all rate matrices (μ_{ij}) satisfying $\sum_i \mu_{ij} \leq 1$, $\sum_j \mu_{ij} \leq 1$. Using Little's Theorem, and the fact that $\sum_{ij} \lambda_{ij} = \rho N$, we have:

$$Delay = \frac{1}{\rho N} \sum_{ij} \overline{L}_{ij} \geq \inf_{(\mu_{ij}) \in \Lambda} \left\{ \frac{1}{\rho N} \sum_{ij} \overline{U}_{ij}(\mu_{ij}) \right\}$$

However, because the $\overline{U}_{ij}(\mu_{ij})$ functions are identical and convex, the expression inside the infimum is a convex symmetric function and attains its minimum at $\mu_{ij} = 1/N$ for all (i, j), and the result follows. □

Note that this lower bound differs by one timeslot from the delay expression in (4) for the periodic scheduling algorithm given in Section II. Because of the $1/(1-\rho)$ factor, delay in the $N \times N$ packet switch with Poisson inputs necessarily grows to infinity as the loading ρ approaches 1. For any fixed loading value, delay grows linearly in the size of the switch. This $O(N)$ result holds more generally. Indeed, consider general stationary, ergodic arrival streams X_{ij} with data rates λ_{ij}, and define the average rate into input ports of the switch to be $\lambda_{av} \triangleq \frac{1}{N} \sum_{ij} \lambda_{ij}$. (Note that in the uniform loading case, $\lambda_{av} = \rho$, and $\lambda_{ij} = \rho/N$.) We assume that there are at least $O(N^2)$ entries of the rate matrix which have rates greater than or equal to $O(\lambda_{av}/N)$.

Theorem 2. *For general stationary, ergodic inputs with data rates λ_{ij}, if $O(N^2)$ of the rates are greater than $O(\lambda_{av}/N)$, then average delay under independent scheduling is at least $O(N)$.*

Proof. The proof is provided in [1]. □

A simple counter-example shows that delay can be $O(1)$ if the rate matrix does not have a sufficient number of entries with large enough rate: Consider a rate matrix equal to the identity matrix multiplied by the scalar $\lambda < 1$. Then, the switch can be configured to always transfer input 1 to output 1, input 2 to output 2, etc., and average delay is the same as the $O(1)$ delay of an output queue.

IV. An $O(\log(N))$ Delay Bound for Backlog-Aware Scheduling

Here we show that $O(\log(N))$ delay is possible by using a backlog-aware scheduling strategy. This result for the first time establishes that sublinear delay is possible in an $N \times N$ packet switch without speedup. The algorithm is similar to the *frame based* schemes considered in terms of stability in [14] [17], and is based on the principle of iteratively clearing backlog in minimum time. Minimum clearance time policies have recently been applied to stabilize *networks* in [22], [16]. We begin by outlining several known results about clearing backlog from a switch in minimum time.

A. Minimum Clearance Time and Maximum Matchings

Consider a single batch of packets present in the switch at time zero. We represent the initial backlog as an *occupancy matrix* (L_{ij}), where entry L_{ij} represents the number of packets at input port i destined for output port j. Suppose that no new packets enter, and the goal is simply to clear all packets in minimum time by switching according to permutation matrices. The following fundamental result from combinatorial mathematics provides the solution to this problem [23]:

Fact 1. *Let T^* represent the minimum time required to clear backlog associated with occupancy matrix (L_{ij}). Then T^* is exactly given by the maximum sum over any row or column of the matrix (L_{ij}).*

It is clear that the minimum time to clear all backlog can be no smaller than the total number of packets in any row or column, because the corresponding input or output can only serve 1 packet at a time. This minimum time can be achieved by an algorithm similar to the Birkhoff-Von Nuemann algorithm described in [2]. Indeed, The matrix is first augmented with *null* *packets* so that every row and column has line sum T^*. Using Hall's Theorem [23], it can be shown that the augmented backlog matrix can be cleared by a sequence of T^* *perfect matches* of size N.

Such matchings can be found sequentially using any Maximum Size Matching algorithm, where each match requires at most $O(N^{2.5})$ operations (see [2] [24] [25] [14]). Note that the preliminary matrix augmentation procedure can be accomplished with $O(N)$ computations each timeslot by updating a set of vectors *row_sum* and *column_sum* each timeslot, and then augmenting the matrix at the beginning of each frame by using these row and column sum vectors to sequentially update each row in the next column which does not have a full sum.

B. Fair Frame Scheduling for Logarithmic Delay

We now present a frame based scheduling algorithm which iteratively clears backlog associated with successive batches of packets. Packets which are not cleared during a frame are marked and handled separately in future frames. The algorithm is "fair" in that when the empirical input rates averaged over a frame are outside of the capacity region of the switch, decisions about which packets to serve are made fairly. We show that if inputs are Poisson and rates are strictly within the capacity region, the switch is stable and yields $O(\log(N))$ delay.

The Fair-Frame Scheduling Algorithm: Timeslots are grouped into frames of size T slots.

1) On the first frame, switching matrices $(S_{ij}(t))$ are chosen randomly so that the probability of serving any particular queue is uniformly $1/N$.
2) On the $(k+1)^{th}$ frame, the backlog matrix $(L_{ij}(kT))$ consisting of packets that arrived during the previous frame is augmented with null packets so that all row and column sums are equal to T. If this cannot be done, an *overflow* occurs. A new matrix $(\tilde{L}_{ij}(kT))$ with row and column sums equal to T is formed from a subset of the backlog of the previous frame. The packets covered by this new matrix will be scheduled on the next frame, and the remaining packets are marked as *overflow* packets. Choice of which $(\tilde{L}_{ij}(kT))$ to use is based upon some type of utility function, such as the FCFS utility or max-min fair utility, described in [1].
3) All non-overflow packets are scheduled during frame $(k+1)$ by performing maximum matches every timeslot to strip off permutations from the augmented backlog matrix.
4) If all packets of the augmented backlog matrix are cleared in less than T slots, uniform and random scheduling is performed on the remaining slots to serve the overflow packets remaining in the system from previous overflow frames. Note that the probability of serving a particular overflow packet at the head of its queue (i, j) during such a slot is $1/N$.
5) Repeat from step 2.

If any packet arriving during a frame k is *not* cleared within the next frame, at least one of the following inequalities must have been violated:

$$\sum_j [X_{ij}((k+1)T) - X_{ij}(kT)] \leq T \text{ for all } i \tag{6}$$

$$\sum_i [X_{ij}((k+1)T) - X_{ij}(kT)] \leq T \text{ for all } j \tag{7}$$

Traffic that satisfies the above inequalities during a frame is said to be *conforming* traffic. Packets remaining in the switch because of a violation of these inequalities are defined as *non-conforming* packets and are served on a best effort basis in future frames. Note that, by definition, the Fair-Frame algorithm clears all conforming traffic within $2T$ timeslots.

Here we describe the performance of the Fair-Frame algorithm with random inputs. Suppose inputs are Poisson with rates λ_{ij} satisfying:

$$\sum_j \lambda_{ij} \leq \rho \text{ for all } i \ , \ \sum_i \lambda_{ij} \leq \rho \text{ for all } j \tag{8}$$

where ρ represents the maximum loading on any input port or output port. Note that if the sum rate to any input or output exceeds the value 1, the switch is necessarily unstable. In the following, we show that if $\rho < 1$, the Fair-Frame algorithm can be designed to ensure stability with delay that grows logarithmically in the size of the switch. We start by presenting a lemma that guarantees that the overflow probability decreases exponentially in the frame length T.

Lemma 2. *For an arbitrarily small overflow probability δ, choose an integer frame size T such that:*

$$T \geq \frac{\log(2N/\delta)}{\log(1/\gamma)} \tag{9}$$

where $\gamma \triangleq \rho e^{1-\rho}$. Then a switch operating under the Fair-Frame algorithm with a frame size T ensures the probability of a frame overflow is less than δ. All conforming packets have a delay less than $2T$, and if $T\sum_{ij}\lambda_{ij} \geq 1$, the fraction of packets which are non-conforming is less than 2δ.

Proof. Packets are lost during frame k only if one of the $2N$ inequalities of (6), (7) is violated during the previous frame. Let $X(T)$ represent the number of packets arriving from a Poisson stream of rate ρ during an interval of T timeslots. Then any individual inequality of (6) and (7) is violated with probability less than or equal to $Pr[X(T) > T]$. By the Chernov bound, we have for any $r > 0$:

$$\begin{aligned} Pr[X(T) > T] &\leq \mathbb{E}\left\{e^{rX(T)}\right\} e^{-rT} \\ &= \exp(\rho T(e^r - 1) - rT) \end{aligned} \tag{10}$$

where the identity $\mathbb{E}\left\{e^{rX(T)}\right\} = \exp(\rho T(e^r - 1))$ was used for the Poisson variable $X(T)$.

To form the tightest bound, define the exponent in (10) as the function $g(r) = \rho T(e^r - 1) - rT$. Taking derivatives reveals that the optimal exponent for the Chernov bound is achieved when $e^r = 1/\rho$. Using this in (10), we have:

$$Pr[X(T) > T] \leq \left[\rho e^{1-\rho}\right]^T$$

We define $\gamma \triangleq \rho e^{1-\rho}$. The γ parameter is an increasing function of ρ for $0 \leq \rho \leq 1$, being strictly less than 1 whenever $\rho < 1$. By the union bound, the probability that any one of the $2N$ inequalities in (6) and (7) is violated is less than or equal to $2N\gamma^T$. Hence, if we ensure that:

$$2N\gamma^T \leq \delta \tag{11}$$

then each frame successfully delivers all of its packets with probability greater than $1 - \delta$. Taking the logarithm of both sides of (11), we obtain the requirement:

$$T \geq \frac{\log(2N/\delta)}{\log(1/\gamma)}$$

Now let $\theta \triangleq T \sum_{ij} \lambda_{ij}$ represent the expected number of packet arrivals during a frame. In [1], we show that for Poisson arrivals, the extra amount of packets that arrive given that the number of arrivals is greater than some value T is stochastically less than the original Poisson variable plus 1. It follows that the expected number of extra arrivals to a frame in which one of the inequalities (6), (7) is violated is less than or equal to $1 + \theta$. Thus, the ratio of non-conforming packets to total packet arrivals is no more that $\delta(1+\theta)/\theta$. Assuming $\theta \geq 1$, it follows that this ratio is less than 2δ. □

The $\log(2N)$ delay bound in (9) arises because of the $2N$ constraints describing the switch capacity region. In a switch with Bernoulli traffic rather than Poisson traffic, no more than one packet can enter any input port. In this case, the constraints in (6) are necessarily satisfied and can be removed from the union bound expression in (11), which reduces the delay bound. Intuitively, a similar argument can be used to prove that logarithmic delay is achievable in any single-hop switching network with a capacity region described by a polynomial number of constraints, as the logarithm of N^k remains $O(\log(N))$.

It is useful to understand how the frame size grows for a fixed overflow probability δ as the loading ρ approaches 1. The formula for the frame size T contains a $\log(1/\gamma)$ term in the denominator. Using the definition of γ and taking a Taylor series expansion about $\rho = 1$ shows that $\log(1/\gamma) = \frac{(1-\rho)^2}{2} + O(1-\rho)^3$. Thus, the denominator is $O((1-\rho)^2)$. This suggests that the cost of achieving $O(\log(N))$ delay is to have a delay which is more sensitive to the loading parameter ρ (confer with eqs. (3)- (5)).

We note that the Poisson assumption is not essential to the proof—a similar proof can be constructed for any independent input streams X_{ij} such that $Pr[\sum_j X_{ij}(T) > T]$ and $Pr[\sum_i X_{ij}(T) > T]$ decreases geometrically with T. It is necessary that the streams be independent for this property to hold. Indeed, consider a situation where all inputs experience the *same* processes, so that $X_{ij}(t) = X(t)$ for all (i,j). Whenever a packet arrives to input 1 destined for output 1, all other inputs receive a packet destined for output 1, and the minimum average delay is $O(N/2)$.

To provide a true delay bound, the delay of non-conforming packets must be accounted for, as accomplished in the theorem below.

Theorem 3. *For Poisson inputs strictly interior to the capacity region with loading no more than ρ, a frame size T can be selected so that the Fair-Frame algorithm ensures logarithmic average delay.*

Proof. For an overflow probability δ (to be chosen later), we choose the frame size $T = \left\lceil \frac{\log(2N/\delta)}{\log(1/\gamma)} \right\rceil$ so that overflows occur with probability less than or equal to δ. The backlog associated with non-conforming packets for any queue (i,j) can be viewed as entering a *virtual G/G/1 queue* with random service opportunities every frame. Let q represent the probability of frame 'underflow': the probability that there is at least one random service opportunity for non-conforming packets during a frame. This is the probability that all backlog of the previous frame can be cleared in less than T slots. Using a Chernov bound argument similar to the one given in the proof of Lemma 2, it can be shown that $Pr[X(T) > T-1] \leq \frac{1}{\rho}\gamma^T$, and hence:

$$q \geq 1 - \frac{2N}{\rho}\gamma^T \tag{12}$$

Expressed in terms of δ, this means that:

$$\begin{aligned} q &\geq 1 - \frac{2N}{\rho}\gamma^{\frac{\log(2N/\delta)}{\log(1/\gamma)}} \\ &= 1 - \frac{2N}{\rho}\gamma^{\log_\gamma(\delta/(2N))} \\ &= 1 - \delta/\rho \end{aligned} \tag{13}$$

The average delay for non-conforming packets in queue (i,j) is thus less than or equal to T (the size of the frame in which they arrived) plus the average delay associated with a slotted G/G/1 queue where a service opportunity arises with probability q/N. Every slot, with probability $1-\delta$ no new packets arrive to this virtual queue (as all packets are conforming), and with probability δ there are $1 + X$ packets that arrive, where X is a Poisson variable with mean ρT (where we again use the result in [1] which shows that excess arrivals are stochastically less than the original). Note that this is a *very large overbound*, as *all* overflow packets arriving to an input i are treated as if they arrived to queue (i,j). Conforming packets consist of at least a fraction $1 - 2\delta$ of the total data and have a delay bounded by $2T$. Thus, the resulting average delay satisfies:

$$\begin{aligned} Delay &\leq 2T(1-2\delta) + 2\delta(T + TDelay(G/G/1)) \\ &\leq 2T + 2\delta TDelay(G/G/1) \end{aligned} \tag{14}$$

where $Delay(G/G/1)$ represents the average delay of non-conforming packets in the virtual G/G/1 queue (normalized to units of frames).

The average delay of a stable, slotted G/G/1 queue with independent arrival and service opportunities can be solved exactly. However, we simplify the exact expression by providing the following upper bound, which is easily calculated using standard queueing theoretic techniques:

$$Delay(G/G/1) \leq \frac{1 + \mathbb{E}\{A^2\}/\lambda}{2(\mu - \lambda)} \quad (\text{for } \mu > \lambda) \tag{15}$$

where, in this context, we have:

$$\lambda = \delta(1 + \rho T) \tag{16}$$

$$\mu = q/N \tag{17}$$

$$\mathbb{E}\left\{A^2\right\} = \delta\mathbb{E}\left\{(1+X)^2\right\} = \delta\left[1 + 3\rho T + \rho^2 T^2\right] \tag{18}$$

The virtual queue is stable provided that $\mu > \lambda$. This is ensured whenever the parameter δ is suitably small. Indeed, we have:

$$\begin{aligned} \mu - \lambda &= \frac{q}{N} - \delta(1+\rho T) \\ &\geq \frac{1}{N} - \frac{\delta}{\rho N} - \delta(1 + \rho T) & (19) \\ &= \frac{1}{N}\left[1 - \delta\left(\frac{1}{\rho} + N + N\rho T\right)\right] & (20) \end{aligned}$$

where inequality (19) follows from (13). Hence, we have $\mu > \lambda$ whenever the following condition is satisfied:

$$\delta\left(\frac{1}{\rho} + N + N\rho T\right) < 1 \tag{21}$$

Choose $\delta = O(1/N^2)$ and note that $T = \lceil\frac{\log(2N/\delta)}{\log(1/\gamma)}\rceil = O(\log(N^3)) = O(\log(N))$. It follows that the left hand side of (21) can be made arbitrarily small for suitably small δ. In particular, we can find a value δ such that $\delta\left(\frac{1}{\rho} + N + N\rho T\right) \leq 1/2$, so that (20) implies $(\mu - \lambda) \geq 1/(2N)$. In this case, we have from (14) and (15) that:

$$\begin{aligned} Delay &\leq 2T + 2\delta T \frac{1 + \mathbb{E}\left\{A^2\right\}/\lambda}{2(\mu - \lambda)} & (22) \\ &\leq 2T + 2\delta T N\left(1 + \frac{1 + 3\rho T + \rho^2 T^2}{1 + \rho T}\right) \\ &= 2T + 2\delta T N\left(1 + \frac{(1+\rho T)^2 + \rho T}{1 + \rho T}\right) \\ &\leq 2T + 2\delta T N\left(2 + 2\rho T\right) \end{aligned}$$

Because $\delta = O(1/N^2)$ and $T = O(\log(N))$, it follows that the resulting average delay is $O(T)$, that is, $Delay \leq O(\log(N))$. □

An explicit delay bound for any N can be obtained for a given loading value ρ as follows: Again define $\gamma \triangleq \rho e^{1-\rho}$, and define the frame size as a function of δ: $T_\delta \triangleq \lceil\frac{\log(2N/\delta)}{\log(1/\gamma)}\rceil$. Using the definitions for λ, μ, and $\mathbb{E}\left\{A^2\right\}$ given in (16)-(20), the average delay bound of (22) can be expressed as a pure function of the parameter δ (as well as the parameter ρ). This bound can be minimized as a function of δ, subject to the constraint that $\delta\left(\frac{1}{\rho} + N + N\rho T_\delta\right) < 1$. The resulting value δ_{min} defines a suitable frame size $T_{\delta_{min}}$ and gives the tightest bound achievable from the above analysis.

In Fig. 2 we plot the resulting delay bound as a function of N for the fixed loading value $\rho = 0.7$. The delay bound for the Fair-Frame algorithm follows a logarithmic profile exactly (the plot is linear when a logarithmic scale is used for the horizontal axis). The bound is plotted next to the exact average delay expressed in (3) for the queue length-independent randomized algorithm.[2] Note the rapid growth in delay as a function of the switch size for the randomized algorithm, as compared to the relatively slow growth for the Fair-Frame algorithm. From the plot, the curves cross when the switch size is approximately 200. However, note that the curve for the Fair-Frame algorithm represents only a simple upper bound, and we conjecture that tighter delay analysis will reveal that the Fair-Frame algorithm is preferable even for much smaller switch sizes.

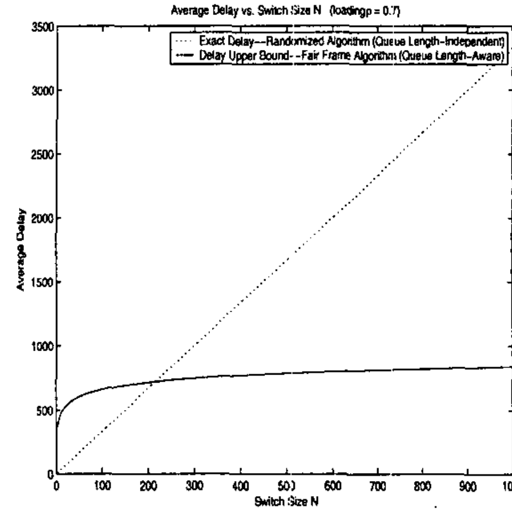

Fig. 2. The logarithmic delay bound for the Fair-Frame algorithm as a function of the switch size N, as compared to the $O(N)$ delay of the randomized algorithm (which was previously the best known delay bound).

We note that although only average delay is compared, the Fair-Frame algorithm has the property that all conforming packets have a worst case delay that is less than or equal to $2T$ (where T is logarithmic in N), and the fraction of conforming packets is at least $1 - O(1/N^2)$. That is, worst case delay is logarithmic for all but a negligible fraction of all packets served.

C. Robustness to Changing Input Rates

Note that the Fair-Frame algorithm requires a loading bound ρ on each input but otherwise does not require knowledge of the exact input rates. For this reason, it can be shown that the Fair-Frame algorithm is *robust to time varying input rates*. Indeed, it is not difficult to show that the Chernov bound of (10) applies even when rates are *arbitrarily changing every timeslot*, provided that on each timeslot the new rates always satisfy the constraints in (8).

In the case when input rates are outside of the capacity region, it is not possible to stabilize the switch. However, the Fair-Frame algorithm makes fair scheduling decisions leading to fair long-term average throughputs in this situation. Indeed, the utility function of the Fair-Frame algorithm can be adjusted

[2] The delay expression (3) for randomized switching algorithms can be shown to hold for any inputs (λ_{ij}) satisfying all inequalities of (8) with equality (see [1]). This bound is almost identical to the bound obtained for the MWM algorithm in [9], and hence the plot in Fig. 2 can also be viewed as a comparison between the MWM bound and the Fair-Frame bound.

to select empirical rates every frame in order to maximum some utility metric, such as the First Come First Served (FCFS) fairness metric, or the max-min fairness metric (see [2]). It is not difficult to show that optimizing over these utility metrics every timeslot leads to a near-optimal long-term througput, where nearness is determined as a function of the frame size.

V. Implementation Complexity

The Fair-Frame algorithm relies on Maximum Size Matchings every timeslot. In [1], it is shown that these matchings act as the implementation complexity bottleneck.[3] In general, maximum size matchings can be performed using the algorithm in [25] which requires $O(MN^{1/2})$ operations, where M is the number of nonzero entries of the backlog matrix. For backlog matrices with many nonzero entries, M can be as large as N^2. However, the Fair-Frame algorithm by definition performs maximum matchings on a backlog matrix for which the total number of packets at any input is no more than T, where T is $O(\log(N))$. It follows that the number of nonzero entries is less than or equal to NT, i.e., M is $O(N\log(N))$. Thus, the Fair-Frame algorithm achieves logarithmic delay and requires $O(N^{1.5}\log(N))$ total operations every timeslot. This $(delay, complexity)$ operating point lies below the delay-complexity curve established for the class of stable algorithms given in [18]. Indeed, in [18] it is shown that for any parameter choice α such that $0 \leq \alpha \leq 3$, a stable scheduling algorithm can be developed requiring $O(N^{\alpha})$ per-timeslot computation complexity and ensuring $O(N^{4-\alpha})$ average delay. Thus, the Fair-Frame algorithm reduces delay by a factor of approximately $O(N^{2.5})$ at the $O(N^{1.5}\log(N))$ complexity level. We conjecture that a new complexity-delay tradeoff curve can be established using the techniques given in [18].

VI. Conclusions

We have considered scheduling in $N \times N$ packet switches with random traffic. It was shown that queue length-independent algorithms, such as those using randomized or periodic schedules designed for known input rates, necessarily incur delay of at least $O(N)$. However, a simple queue length-aware algorithm was constructed and shown to provide delay of $O(\log(N))$. This is the first analytical demonstration that sublinear delay is possible in a packet switch, and proves that high quality packet switching with the crossbar architecture is feasible even for very large switches of size $N > 1000$. The Fair-Frame algorithm provided here is based on well established framing techniques and is simple to implement, requiring only $O(N^{1.5}\log(N))$ computations every timeslot. Although the logarithmic delay proof was performed using the Poisson input assumption, it is intuitively clear that similar results apply for more general input streams.

Performance of Fair-Frame can likely be improved by enabling dynamic frame sizing and alternative matching assignments. An important question for future research is that of developing delay-optimal scheduling. Such scheduling would yield delay which is upper bounded by $O(\log(N))$ and lower bounded by $O(1)$, which now serve as the tightest known bounds on optimal delay.

[3] Indeed, it is shown in [1] that, for simple utility metrics, "Step 2" of the Fair-Frame algorithm can be performed using fewer than $O(N^{1.5})$ operations per timeslot.

References

[1] M. J. Neely and E. Modiano. Logarithmic delay for $n \times n$ packet switches. *USC CSI Technical Report: CSI-04-01*, February 2004.

[2] C-S Chang, W-J Chen, and H-Y Huang. Birkhoff-von neumann input buffered crossbar switches. *IEEE Proceedings of Infocom*, 2000.

[3] C-S Chang, D-S Lee, and C-Y Yue. Providing guaranteed rate services in the load balanced birkhoff-von neumann switchies. *IEEE Proceedings of INFOCOM*, April 2003.

[4] E. Leonardi, M. Mellia, F. Neri, and M. A. Marsan. On the stability of input-queued switches with speedup. *IEEE/ACM Transactions on Networking*, Vol. 9, n.1, Feb. 2001.

[5] C.E. Koksal. *Providing Quality of Service over Electronic and Optical Switches.* PhD thesis, Massachusetts Institute of Technology, Laboratory for Information and Decision Systems (LIDS), Sept. 2002.

[6] M. Andrews and M. Vojnović. Scheduling reserved traffic in input-queued switches: New delay bounds via probabilistic techniques. *IEEE Proceedings of INFOCOM*, April 2003.

[7] N. McKeown, V. Anantharam, and J. Walrand. Achieving 100% throughput in an input-queued switch. *Proceedings of IEEE INFOCOM, pp. 296-302*, 1996.

[8] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transacations on Automatic Control*, Vol. 37, no. 12, Dec. 1992.

[9] E. Leonardi, M. Melia, F. Neri, and M. Ajmone Marson. Bounds on average delays and queue size averages and variances in input-queued cell-based switches. *Proceedings of IEEE INFOCOM*, 2001.

[10] S-T Chuang, A. Goel, N. McKeown, and B. Prabhakar. Matching output queueing with a combined input output queued swtich. *IEEE Proceedings of Infocom*, 1998.

[11] S. Sarkar. Optimum scheduling and memory management in input queued switches with finite buffer space. *IEEE Proceedings of INFOCOM*, April 2003.

[12] M. J. Neely, E. Modiano, and C. E. Rohrs. Dynamic routing to parallel time-varying queues with applications to satellite and wireless networks. *Proceedings of the Conference on Information Sciences and Systems*, Princeton: March 2002.

[13] M. J. Neely. *Dynamic Power Allocation and Routing for Satellite and Wireless Networks with Time Varying Channels.* PhD thesis, Massachusetts Institute of Technology, Laboratory for Information and Decision Systems (LIDS), 2003.

[14] T. Weller and B. Hajek. Scheduling nonuniform traffic in a packet switching system with small propagation delay. *IEEE/ACM Transactions on Networking*, 5 (6), 588-598, 1998.

[15] M. Andrews and L. Zhang. Achieving stability in networks of input-queued switches. *IEEE Proceedings of INFOCOM*, 2001.

[16] M. J. Neely, J. Sun, and E. Modiano. Delay and complexity tradeoffs for routing and power allocation in a wireless network. *Allerton Conference on Communication, Control, and Computing*, Oct. 2002.

[17] S. Iyer and N. McKeown. Maximum size matchings and input queued switches. *Proceedings of the 40th Annual Allerton Conference on Communication, Control, and Computing*, October 2002.

[18] M. J. Neely, E. Modiano, and C. E. Rohrs. Tradeoffs in delay guarantees and computation complexity for $n \times n$ packet switches. *Proceedings of the Conference on Information Sciences and Systems, Princeton University*, March 2002.

[19] D. Shah and M. Kopikare. Delay bounds for the approximate maximum weight matching algorithm for input queued switches. *IEEE Proceedings of INFOCOM*, 2002.

[20] D. P. Bertsekas and R. Gallager. *Data Networks.* New Jersey: Prentice-Hall, Inc., 1992.

[21] P. Humblet. Determinism minimizes waiting time. *MIT LIDS Technical Report P-1207*, May 1982.

[22] D. Gamarnik. Stability of adaptive and non-adaptive packet routing policies in adversarial queueing networks. *Proc. of 31st ACM Symposium on the Theory of Computing*, 1999.

[23] M. Hall Jr. *Combinatorial Theory.* Waltham, MA: Blaisedell, 1969.

[24] C.H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithm and Complexity.* New Jersey: Prentice Hall, 1982.

[25] J. Hopcroft and R. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM J. Comput.*, Dec. 1973.