Worst-Case Delay Bounds for Uniform Load-Balanced Switch Fabrics

Spyridon Antonakopoulos, Steven Fortune, Rae McLellan, Lisa Zhang
Bell Laboratories, 600 Mountain Ave, Murray Hill, NJ 07974
firstname.lastname@alcatel-lucent.com

***Abstract*—Numerous solutions have been proposed in the literature to eliminate reordering in load-balanced switch fabrics. A common approach involves uniform frames, in which every cell of a frame has the same destination. This can achieve 100% throughput with relatively small average traffic delay; however, the worst-case delay may be unbounded. We show that with a slight speedup in the switch fabric we can guarantee satisfactory worst-case delay bounds, without sacrificing other desirable properties. Furthermore, experimental results demonstrate that our scheme improves worst-case delay in realistic traffic scenarios, as compared to previous uniform-frame solutions.**

I. Introduction

Load balancing in switch fabrics has been studied extensively for more than a decade, as a building block to obtain high-capacity routers; see e.g. [1], [2], [3], [7], [8], [9], [12], among others. The primary motivation behind these research efforts is the explosive growth in Internet traffic, variously estimated at 30-50% per year [4], [10]. As a result of that trend, load-balanced switch fabrics have become increasingly important because they are inherently parallel, enabling switch capacity to scale even without any increase in VLSI clock rate. Furthermore, load-balanced switch fabrics are known to attain 100% throughput for a reasonably general model of admissible traffic [1], [2].

A typical load-balanced switch fabric consists of three stages of cell-handling elements. In this paper, we focus on frame-based fabrics with a full-mesh interconnect between successive stages, as illustrated in Figure 1. Data packets, already split into fixed-size cells, arrive at a first-stage *distributor*. Cells are accumulated into a frame of cells that are sent out in parallel to the middle stage (called the *routing stage*), one cell per *routing element*. Each routing element similarly accumulates frames of cells to be sent out in parallel to the final *collector* stage. Nevertheless, unlike a distributor, each routing stage performs routing, so that each cell is sent to its intended collector. From there cells exit the switch fabric and are then reassembled into packets.

Frame-based fabrics allow high-aggregate-capacity switches to be built from elements that need not operate much faster than the rate at which traffic can arrive at a distributor. Moreover, if there are m elements at each stage, each link between stages operates at a fraction $1/m$ of the rate into the distributor. Ideally, the m^2 links between stages could be encapsulated in a single optical device, a polycyclic array wave guide [9]. We remark that load-balanced switch fabrics were initially described with crossbar-based interconnects realizing a fixed schedule of matchings [1], [2]; the full mesh described above replaces the crossbar, which would be challenging to implement at high rates.

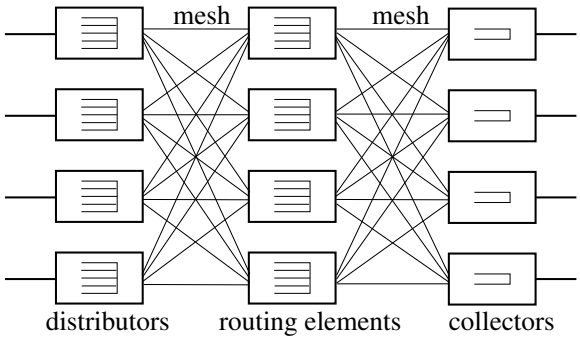

Fig. 1. Frame-based switch fabric

Cells from a specific distributor sent to a specific collector may traverse different routing elements. If these routing elements do not impose the same delay (e.g. because of differing queue occupancies), cells are likely to arrive at the collector in a different order from which they were sent. Requiring that frames be *uniform*, i.e. all cells in each frame have the same destination, has been proposed as a way to prevent this cell-reordering problem [7], [8]. Indeed, uniform frames guarantee that, at any given time and for any given collector, all routing element queues to that collector have identical occupancies and thus incur identical delays. In this way, cell reordering absolutely cannot occur, since each frame arrives at its destined collector as a whole, exactly as it was sent from the distributor, and of course no frame can overtake an earlier frame sent to the same collector.

Several switch scheduling policies that are based on the aforementioned approach [7], [8] provably achieve 100% throughput and ensure bounded average cell delay. Unfortunately, though, there is no guarantee whatsoever in terms of the worst-case delay, which can be arbitrarily large. Providing such a guarantee is the main goal of this paper.

More specifically, we present the *Bounded Latency Frames (BLF)* scheduling policy, which permits a frame to be sent from a distributor if: (a) there are enough cells with the same destination to fill a frame; or (b) some cell in the distributor has been waiting for more than a specific period of time. We subsequently show that, for any traffic arrival pattern, a uniform-frame-based switch fabric with this scheduling policy

delays a cell by at most a constant more than would an ideal output-queued switch.

To obtain the above result, we require that the switch fabric have a slight speedup, because sending frames before they are completely filled wastes bandwidth. Additionally, we assume that internal switch queues do not have maximum occupancy limits and may grow arbitrarily large—which is standard for such theoretical comparisons. Therefore, our result complements those of [7], [8]: a stronger assumption (namely, fabric speedup) yields a stronger guarantee (i.e., worst-case bounded delay rather than average-case bounded delay).

The paper is organized as follows. Section II describes our BLF scheduling policy and formally states the performance guarantee that BLF provides. All technical proofs, however, are relegated to Section III. In Section IV we evaluate the performance of BLF in realistic settings via simulations. Last but not least, Section V presents our concluding remarks.

II. Bounded Latency Frames

A. Preliminaries

As mentioned earlier, in a *uniform* load-balanced switch, every frame sent by a distributor is uniform, that is, every cell in the frame has the same collector as destination. Consider the schematic representation of Figure 1, and let m denote the number of elements on each of the three stages (distributors, routing elements, and collectors). Then, each frame consists of exactly m cells.

At fixed intervals, each of which is called a *frame time*, every distributor sends a frame to the routing stage. For that purpose, every distributor must apply the scheduling policy independently to determine the collector for which each frame is intended. If necessary, *empty* cells containing dummy data may be added to a frame to reach the requisite number of m cells so that it can be sent.

Similarly, at each frame time, every routing element sends at most one cell to each collector. Routing elements operate deterministically, all using the same algorithm to fill queues. For example, if at a given frame time multiple distributors send frames destined to the same collector, then the order in which the routing elements' queues to that collector are filled with cells from those frames is identical for all elements (e.g. break ties by the index of each frame's distributor). Consequently, at any time, all the routing-stage queues associated with a specific collector have the same length, and furthermore all cells associated with a particular distributor frame are at the same position in these queues. Thus, all cells in a frame arrive at their destination simultaneously.

Note that an empty cell used to fill out a frame occupies a position within a routing element queue, and persists until it reaches a collector, where it is dropped. As an exception to the above, a frame consisting only of empty cells[1] should be marked by the distributor so that it can be discarded immediately after arriving at the routing stage.

[1]Sending such frames seems pointless, but in practice it could be required to maintain fabric synchronization.

Finally, we assume that during each frame time at most

$$r = \left\lfloor \frac{m}{\alpha} \right\rfloor$$

cells may arrive at a distributor, and also at most r cells may depart a collector. The quantity $\alpha > 1$ expresses the *speedup* of the switch fabric compared to the capacity of the input and output links at the distributors and collectors, respectively.

B. Scheduling policy

A very obvious performance concern in uniform switch fabrics is the delay incurred while a distributor waits for enough cells with the same destination to create a frame. Filling up frames with empty cells enables them to be sent in a more timely fashion, but doing this too often consumes a lot of bandwidth and leads to high queueing delays in the routing stage. Nevertheless, by applying the above technique judiciously and exploiting the speedup α, we demonstrate a scheduling policy that bounds worst-case delay.

As cells arrive at a distributor, they are placed in the appropriate virtual output queue by destination, with the oldest cell at the head of the queue. At any given time, the cells in a queue are partitioned in order from oldest to youngest into (potential) frames: zero or more *full* frames containing exactly m cells, and possibly a *partial* frame containing at least one but fewer than m cells. Now, define

$$\Delta = \left\lceil \frac{\alpha m}{\alpha - 1} \right\rceil . \tag{1}$$

A partial frame is *stale* if the current frame time exceeds the arrival time of the oldest cell in the frame by at least Δ.

In the *Bounded Latency Frames (BLF)* scheduling policy, each distributor examines its virtual output queues in round-robin order, choosing a queue to serve if it contains either a full frame or a stale frame. Thus, the frame at the head of the chosen queue is sent to the routing stage, padded with empty cells if it is stale. If no queue has a full or stale frame, a frame of all empty cells is sent instead.

Lemma 1. *Under the BLF scheduling policy, no cell is delayed at a distributor for more than $\Delta + m$ frame times.*

We have already pointed out that sending stale frames squanders bandwidth. To establish that the speedup α suffices to make up for that loss, we relate the delay experienced by an arbitrary cell through the uniform switch fabric with that of an ideal output-queued switch. In the latter, at each frame time, any cells that arrive at the distributors are instantaneously forwarded to the appropriate collector, and then r cells depart from each collector queue—or the entire queue, if its occupancy is less than r.

An *arrival sequence* $\{a_t\}$ (for a specific collector) indicates the total number a_t of cells arriving over all distributors destined for the collector during the t-th frame interval. The *intrinsic backlog* b_t (for a specific collector) at time t for an arrival sequence $\{a_t\}$ is defined as

$$b_t = \max\left(0,\ \max_{0 \le f \le t} \sum_{i=0}^{f} (a_{t-i} - r)\right) .$$

It is easy to see that b_t expresses the number of cells in the collector queue of an ideal output-queued switch at time t, after the r cells have been emitted. This can also be verified by an inductive argument: it is obviously true at time 0, and at time t the number of cells in the queue is $\max(0, b_{t-1}+a_t-r)$, which equals b_t. Moreover, we define the *intrinsic delay* δ_t at time t as

$$\delta_t = \left\lceil \frac{b_t}{r} \right\rceil .$$

Clearly, δ_t is the delay experienced by the last cell to arrive at the output-queued switch during frame interval t.

Theorem 2. *Suppose a uniform switch fabric with the BLF scheduling policy has unbounded internal queues. Then, a cell arriving at a distributor at time t will exit the collector within $\delta_{t+\Delta+m} + 2\Delta + 3m$ frame intervals.*

In other words, for any arrival sequence, the delay seen by a cell traversing the uniform switch fabric is only up to a constant (depending on m and α) greater than the delay experienced by a cell arriving at almost the same time at an ideal output-queued switch.

Remark. In contrast to the above, suppose the switch has a congestion control mechanism that prevents internal queues from growing too large by temporarily forcing upstream elements to stop sending more cells. In such a scenario, the delay guarantees of Lemma 1 and Theorem 2 are no longer valid—even for traffic to a collector that is never congested. This complication is *not* specific to the uniform load-balanced switch, and additional speedup is required to address it, even with cross-bar-based fabrics [5].

C. Extensions

Consider a situation in which the incoming traffic load to the switch is low, i.e. on average much fewer than r cells arrive at each distributor per frame time. Under the BLF scheduling policy, these cells will experience a delay of roughly Δ frame times, in expectation. This leaves room for improvement, since the bandwidth of the switch is underutilized.

In particular, we can reduce average cell delay in the aforementioned setting by combining BLF with a technique inspired from [7]. Namely, if a distributor has no virtual output queue with a full or stale frame to serve, it again examines the queues in round-robin order, choosing a queue Q to serve if: (a) Q contains a partial frame; and (b) the routing-stage queues associated with the same collector as Q currently hold less than T frames. Here, T is a parameter that we set as desired. Naturally, if there is still no suitable queue to serve, a frame of all empty cells is sent.

As a result, this modified BLF policy tends to decrease the time that cells have to wait at a distributor, provided that the routing-stage queues they will be sent to have sufficiently low occupancies. More importantly, the worst-case delay guarantee of Theorem 2 is only minimally affected by that change.

Theorem 3. *Suppose a uniform switch fabric with the (modified) BLF scheduling policy has unbounded internal queues. Then, a cell arriving at a distributor at time t will exit the collector within $\delta_{t+\Delta+m} + 2\Delta + 3m + T$ frame intervals.*

III. Proofs

Proof of Lemma 1: We say that a frame in a distributor queue is *blocked* if there is a frame ahead of it in the queue, necessarily a full frame. Now, observe that the lemma is straightforward for any frame that is not blocked, since at worst it will become stale in Δ frame times, and will then be served within another m frame times.

So, let us assume that some frame is blocked at frame time $t+1$ (by the arrival of the $(m+1)$-st cell to a queue), but there were no blocked frames at time t. We claim that at some frame time between $t+1$ and $t+\Delta$, inclusive, there will be no full frames, hence no blocked frames. Every once-blocked frame will either have been served or become head of its queue; in the latter case, the frame will be served within m time units of becoming full or stale.

To see the claim, first note that there must exist a full frame at every frame time that there is a blocked frame, which implies that some frame is served then. In the frame times between $t+1$ and $t+\delta$, $\delta \le \Delta$, there are clearly δ service opportunities; how many frames will need to be served? We argue at most $m + \lfloor \delta/\alpha \rfloor$.

Indeed, there are at most m frames that are partial or full at frame time t, one per queue, but no blocked frames. Moreover, if a partial frame is served because it became stale, no other stale frame from the same queue is served during that interval. As a result, any other frame that is served must have all its cells arrive between frame times t and $t+\delta$. There can be at most $\delta r = \delta \lfloor m/\alpha \rfloor$ such cells, hence at most $\lfloor \delta/\alpha \rfloor$ full frames. Now, since

$$m + \lfloor \delta/\alpha \rfloor \le (\alpha - 1)\Delta/\alpha + \Delta/\alpha \le \Delta ,$$

there exists some $\delta \le \Delta$ such that, by frame time $t+\delta$, there will have been as many service opportunities as frames to serve, and all full frames will have been served. ∎

For the following proofs, we only count queueing delays and ignore any delays due to processing or transfers between stages. Thus, a cell that arrives at a distributor at frame time t could be emitted from a collector also at time t if it were not delayed because of queue occupancy.

A collector queue is *backlogged* at time t if it is not empty after the departure of the r cells at time t. Suppose a cell is emitted from a collector queue at time t. The *collector backlog interval* of the cell is the interval $[s..t-1]$ such that the queue is backlogged at times $s, \ldots, t-1$, but not backlogged at time $s-1$. Likewise, the routing-stage queues for a specific collector are *backlogged* at time t if they contain another frame to be sent to the collector (containing at least one nonempty cell). Furthermore, the *routing backlog interval* of a cell is defined similarly to the collector backlog interval.

Proposition 4. *Suppose the routing-stage queues for a specific collector are backlogged for an interval I of g consecutive frame times, but not backlogged before I. Then, the frames sent*

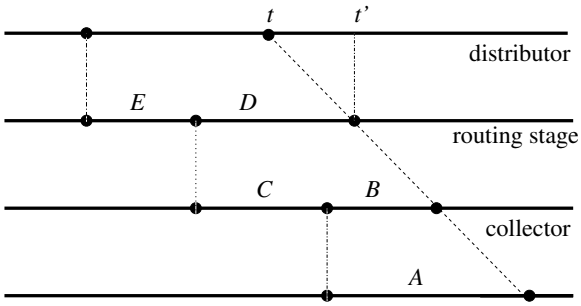

Fig. 2. Time lines (horizontal) in order from top to bottom: arrival time at distributor; departure from distributor/arrival at routing stage; departure from routing stage/arrival at collector; departure from collector. The diagonal dotted line represents the trajectory of cell c arriving at time t.

to the collector during I contain at least $r(g-m)$ nonempty cells.

Proof: Since the routing stages were not backlogged before I, any frames sent to the collector must have arrived during I. Write $g = g_1\Delta + g_2$, where $g_1 = \lfloor g/\Delta \rfloor$ and $g_2 = g \bmod \Delta$. At most $g_1 + 1$ stale frames can arrive at the routing stage during the interval from a single distributor. However, there can be at most $\min(g_2, m)$ such distributors, since the arrival times at the routing stage of the first and last stale frames are separated by at least $g_1\Delta$. Other distributors must contribute g_1 or fewer stale frames each. Hence, the total number of stale frames summed over all distributors is $\le g_1 m + \min(g_2, m)$.

To finish the proof, the number η of nonempty cells is at least m times the number of full frames:

$$\begin{aligned}\eta &\ge m\Big(g - \big(g_1 m + \min(g_2, m)\big)\Big)\\ &\ge r\alpha\left(g_1\frac{m\alpha}{\alpha-1} + g_2 - \frac{g_1 m(\alpha-1)}{\alpha-1} - \min(g_2, m)\right)\\ &= r\left(\frac{g_1 m\alpha}{\alpha-1} + g_2 + (\alpha-1)g_2 - \alpha\min(g_2, m)\right)\\ &\ge r(g-m)\,,\end{aligned}$$

where the inequality $(\alpha-1)g_2 - \alpha\min(g_2, m) \ge -m$ is easily verified using a case split on $g_2 \le m$ or $g_2 > m$. ∎

Proof of Theorem 2: Consider cell c arriving at a distributor at time t, destined for a specific collector. Define intervals of frame times A, B, C, D, and E as follows. Let $A = [s_A..t_A - 1]$ be the collector backlog interval of c. $B \subseteq A$ is the interval from the start of A, i.e. s_A, to the arrival time of c at the collector. Obviously, at time s_A one or more cells must arrive at this collector from the routing stage, otherwise the collector would not become backlogged. The routing backlog interval of these cells is C, which may be empty. $D \subseteq B \cup C$ is the interval from the start of C (or of B, if C empty) to the arrival of c at the routing stage. Finally, E is the interval of length $\Delta + m$ before D. Figure 2 provides a visualization of the above definitions.

We wish to bound the number a of cells that are emitted from the collector during A. Any such cell must have arrived at the collector during B, since A is the backlog interval of c. Likewise, any cell that arrives at the collector during $B \cup C$ must have arrived at the routing stage during D. By Proposition 4, the number of cells sent to the collector during C is at least $r(|C| - m)$, where $|C|$ is the length of C. By Lemma 1, any cell sent to the routing stage during D must have arrived at the distributor during $D \cup E$. Consequently,

$$a \le \sum_{s \in D \cup E} a_s - r(|C| - m).$$

We have $|A| = \lceil a/r \rceil$ since A is the backlog interval for c. Let t' be the endpoint of D. Using $|E| = \Delta + m$,

$$\begin{aligned}|A| + |C| - |D| &\le \left\lceil \frac{1}{r}\sum_{s \in D \cup E} a_s \right\rceil + m - |D|\\ &= \left\lceil \frac{1}{r}\sum_{s \in D \cup E} (a_s - r) \right\rceil + \Delta + 2m\\ &\le \delta_{t'} + \Delta + 2m\,.\end{aligned}$$

The cell is emitted at time

$$\begin{aligned}t_A &= t + (t' - t) + |A| + |C| - |D|\\ &\le t + (t' - t + \delta_{t'}) + \Delta + 2m\\ &\le t + (\Delta + m + \delta_{t+\Delta+m}) + \Delta + 2m\,,\end{aligned}$$

where the second inequality uses the observation that the quantity $t^* - t + \delta_{t^*}$ cannot decrease as t^* varies from t to $t + \Delta + m$. ∎

Sketch of proof of Theorem 3: The proof is very similar to that of Theorem 2, so here we only highlight their differences.

To begin with, observe that Lemma 1 holds for the modified BLF policy as well. Moreover, let us amend two definitions from Section III. In particular, the routing-stage queues for a specific collector are backlogged at time t if they contain more than T frames, and the routing backlog interval of a cell is analogously redefined. Under the new definitions, Proposition 4 also carries over, with a minor change in its statement: the frames sent to the collector during $I + T$ (i.e. the interval I shifted forward by T frame times) contain at least $r(g - m)$ nonempty cells.

Lastly, for the arguments in the proof of Theorem 2 to go through, we modify the intervals B and C: B now starts at time $s_A - T$ instead of s_A, and C is the backlog interval of the cells, if any, that at time $s_A - T$ occupy the T-th position in the routing-stage queues for the collector in question (which means that they will be sent to the collector at time s_A). It is easy to deduce that the net result of all those changes is simply an additional term T in the delay bound. ∎

IV. Simulations

To complement the theoretical analysis, we conducted performance simulations of the BLF scheduling policy, specifically its version described in Section II-C, under realistic traffic patterns. The benchmark used for these simulations is the closely related *Padded Frames (PF)* policy [7], which has

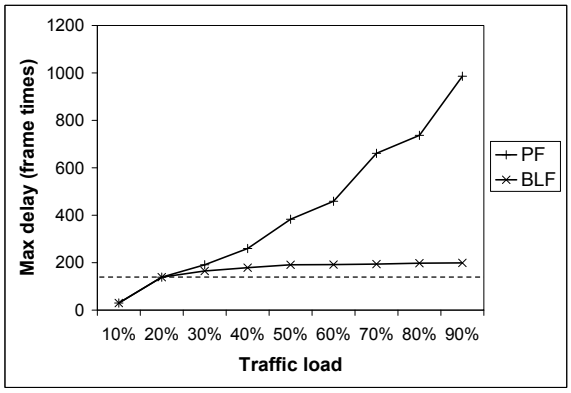

Fig. 3. Plot of maximum packet delay versus traffic load for the PF and BLF scheduling policies, on a 32×32 switch. The horizontal dotted line shows the value of Δ, included for reference purposes.

been experimentally shown to exhibit lower average delay than other proposed uniform-frame policies.

The load-balanced switch fabric on which PF and BLF were applied had $m = 32$ elements per stage, and a small speedup $\alpha = 1.28$. Thus, (1) yields $\Delta = 147$. Additionally, the parameter T was set to 4 for both policies. Note that PF does not require any speedup, unlike BLF; nevertheless, for the purpose of a fair comparison we chose to keep the speedup in both cases. Furthermore, the incoming traffic at distributor i destined for collector j follows a Pareto distribution with average rate ρ_{ij}. These rates are randomly chosen for each distributor-collector pair, subject to the condition

$$\sum_i \rho_{ij} = \sum_j \rho_{ij} = \ell \,,$$

for all i and j. The parameter ℓ above, which is independent of i and j, represents the average *traffic load* on the switch, and in our test instances it ranges from 10% to 90% of the capacity of each of the input and output links that carry traffic into the distributors and out of the collectors, respectively.

Figure 3 presents the maximum observed delay that was incurred under each scheduling policy, expressed in frame times. As expected, BLF ensures that all traffic traverses the switch in a timely manner. By contrast, for moderate to high loads, PF causes some packets to experience very large delays.

It is also instructive to examine these results in conjunction with the corresponding average delays, which are displayed in Table I. Note that both policies attain almost identical average delays in each case, with BLF gaining a very slight advantage as the load increases. Consequently, it appears that BLF's focus on bounding maximum delay (by expediently sending stale frames, which consume more bandwidth) does not harm overall performance in terms of average delay—on the contrary, it marginally improves it.

TABLE I
AVERAGE PACKET DELAY FOR PF AND BLF

Traffic load	PF	BLF
10%	3.64	3.64
20%	7.69	7.69
30%	23.73	23.73
40%	29.42	29.42
50%	31.44	31.44
60%	32.34	32.34
70%	32.84	32.82
80%	33.16	33.09
90%	33.44	33.20

V. CONCLUSION

In this paper, we introduced a novel scheduling policy for load-balanced switch fabrics, called Bounded Latency Frames. We also proved that, under the BLF policy, the worst-case delay experienced by traffic passing through the switch is at most a constant more than the corresponding delay through an ideal output-queued switch; this distinguishes BLF from previously known uniform-frame policies. Moreover, in order to be implemented, BLF requires only a small speedup in the switch fabric, e.g. in the order of 1.28. Last but not least, experimental evidence indicates that in practice the maximum delay bound that BLF achieves does not come at the expense of worse average delay, whence we conclude that BLF delivers a well-rounded performance.

REFERENCES

[1] C. Chang, D. Lee, Y. Jou, Load-balanced Birkhoff-von Neumann switches: Part I: One-stage Buffering, *Comput. Commun.* **25**:6, 2002, pp. 611–622.

[2] C. Chang, D. Lee, Y. Jou, Load-balanced Birkhoff-von Neumann switches: Part II: Multi-stage Buffering, *Comput. Commun.* **25**:6, 2002, pp. 623–634.

[3] C. Chang, D. Lee, Y. Shih, C. Yu, Mailbox Switch: A Scalable two-Stage Switch Architecture for Conflict Resolution of Ordered Packets, *IEEE Transactions on Communications*, **56**:1, January 2008, pp. 136–149.

[4] CISCO Visual Networking Index: Forecase and Methodology, 2011-2016, available within www.cisco.com.

[5] S.-T. Chuang, A. Goel, N. McKeown, B. Prabhakar, Matching Output Queueing with a Combined Input Output Queued Switch, Computer Systems Technical Report CSL-TR-98-758. March 1998, or *Proceedings of INFOCOM '99*, 1169–1178, IEEE, April 1999, or *IEEE Journal on Selected Areas in Communications*, **17**:6, December 1999, pp. 1030-1039.

[6] W. Feller, *An Introduction to Probability Theory and its Applications*, Volume I, Third Edition, John Wiley and Sons, NY, 1950.

[7] J. Jaramillo, F. Milan, R. Srikant, Padded Frames: a Novel Algorithm for Stable Scheduling in Load-Balanced Switches, *IEEE/ACM Transactions on Networking*, **16**:5, October 2008, pp. 1212–1225.

[8] I. Keslassy, The Load-balanced Router, Ph.D. Dissertation, Stanford Univ, Stanford CA, 2004.

[9] I. Keslassy, S.-T. Chuang, K. Yu, D. Miller, M. Horowitz, O. Solgaard, N. McKeown, Scaling Internet Routers Using Optics (Extended Version), Stanford Technical Report TR03-HPNG-080101, 2003.

[10] Minnesota Internet Traffic Studies (MINTS), available at `http://www.dtc.umn.edu/mints/home.php`.

[11] M. Mitzenmacher, E. Upfal, *Probability and Computing: Randomized algorithms and probabilistic analysis*, Cambridge University Press, Cambridge, 2005.

[12] Y. Shen, S. Panwar, H. Chao, Design and Performance Analysis of a Practical Load-Balanced Switch, *IEEE Transactions on Communications* **57**:8, August 2009, pp. 2420–2429.