A Scalable Frame-based Multi-Crosspoint Packet Switching Architecture

Xike Li, *Student Member*, Itamar Elhanany, *Senior Member*
Department of Electrical & Computer Engineering
The University of Tennessee
Knoxville, TN 37996-2100

***Abstract*— Input-queued cell switches employing the oldest-cell-first (OCF) policy have been shown to yield low mean delay characteristics. Moreover, it has been proven that OCF is stable for admissible i.i.d. arrival traffic when executed with a scheduling speedup of 2. However, an increase in link rates and port densities directly leads to a decrease in packet duration times, to a point where cell-by-cell switching is no longer considered practical. To address this challenge, this paper studies frame-based scheduling algorithms for a scalable combined input-output queued (CIOQ) switch architecture. The latter is decomposed into independent subgroups, each employing multiple simple crosspoint switches. A key outcome of this decomposition is a substantial reduction of scheduling times. Unlike many other schemes, which necessitate custom integrated circuits, the architecture proposed here utilizes commercially available crosspoint switches. We present a Lyapunov-based stability analysis that dictates moderate conditions under which the switch is stable for all admissible traffic patterns. By reconfiguring the crossbar switch once every several time slots, the timing constraints imposed on the scheduling algorithm are significantly relaxed. Simulation results are presented, demonstrating the merits of the approach, particularly in the presence of bursty traffic scenarios.**

I. Introduction

Recently, several novel architectures have been proposed for the design of packet switches with large aggregate capacities and high-speed link data rates. Examples include the Parallel Packet Switch (PPS) [1], the Parallel Shared Memory (PSM) router [2], and the Load-Balanced router [3]. In theory, both the PPS and the PSM can emulate a first-come-first-served output-queued (FCFS-OQ) packet switch and support quality of service (QoS) guarantees. Nevertheless, implementation of PPS and PSM switches involves the design of intricate centralized schedulers, which inherently introduce scalability limitations. It has been shown that the Load-Balancing architecture can guarantee 100% throughput for a broad class of traffic patterns and requires no scheduler. However, the Load-Balancing architecture suffers from packet reordering, a consequence of allowing multiple internal input-to-output paths for each flow. Moreover, it imposes frequent switch fabric reconfigurations [4]. These two attributes introduce delay and scalability limitations, and consequently require the introduction of custom-designed VLSI components.

Input-queued (IQ) packet switching architectures with virtual output queueing (VOQ) are commonly utilized in Internet routers as they offer pragmatic scalability while requiring moderate memory bandwidth. A scheduling algorithm is needed for an IQ switch to dynamically determine the configuration of the crossbar by finding matchings between ingress and egress ports. However, an increase in link rates directly causes a decrease in packet duration times to a point where cell-by-cell switching is no longer considered a practical approach. This is true particularly for optical switch fabrics that employ slowly reconfiguring crossconnect elements. The reconfiguration overhead for a typical optical switch fabric can be in the range of 50-100ns [4]. However, with 64-byte packets and speeds of 40 Gbps, a reconfiguration time of a few nanoseconds is necessary for the cell-by-cell switching mechanism. To address this issue, in a previous paper [5], the authors have proposed a frame-based maximal weight matching (FMWM) algorithm with *transfer speedup*, in which scheduling decisions are issued in accordance with the MWM algorithm, however they are kept unchanged for a duration of k consecutive time slots. It has been proven that a CIOQ switch running the FMWM scheduling algorithm with a transfer speedup of 2 is stable under admissible traffic for any frame size. By reconfiguring the crossbar switch once every several time slots, we significantly relax the timing constraints imposed on the scheduling algorithm.

In order to scale to high port densities, we propose a novel scalable packet switching architecture which is straightforward to implement, as it employs a group of memoryless passive crosspoint switches. The architecture is a CIOQ switch whereby an $N \times N$ switch is partitioned into G identical and independent switching groups, each hosting a pool of smaller crosspoint switches. The motivation to employ such an architecture is to benefit from the idea of partitioning one (large) crossbar into several small crosspoints [6], so as to facilitate scalability and reduce the timing requirements from the scheduling algorithm. The approach is characterized by offering a 100% throughput guarantee for a broad class of traffic patterns when employing the FMWM/OCF scheduling algorithm. Packet reordering need not be considered and switch fabric reconfiguration is infrequent. Since the algorithm studied is frame-based, by reconfiguring the crossbar switch once every several time slots, the timing constraints imposed on crosspoint devices are significantly relaxed.

II. General Switch Architecture

The proposed switch architecture is based on a design first introduced in [6]. Consider an $N \times N$ switch as shown in

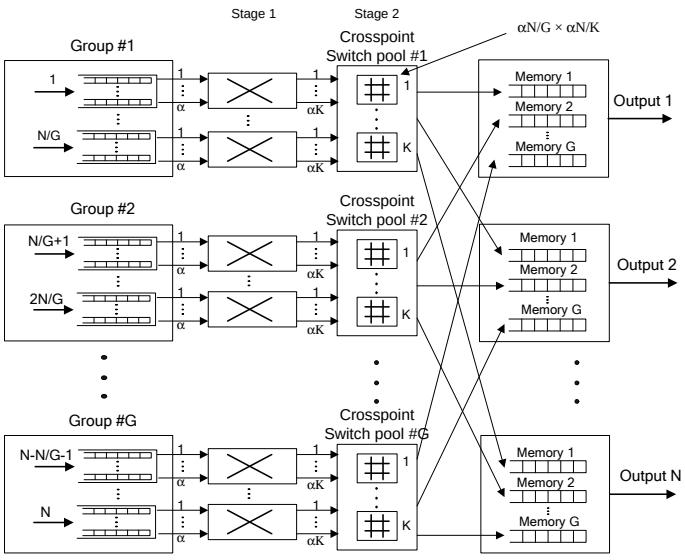

Fig. 1. Proposed switch architecture in which an $N \times N$ CIOQ switch is equally partioned into G independent groups employing multiple passive crosspoints switches.

figure 1, whereby N input modules are equally partitioned into G groups, each of which is independently connected to all N outputs via a pool of non-blocking crosspoint switches. It is assumed that each of the crosspoint switches is small to facilitate practical scalable switch implementations.

Throughout this paper we shall refer to a *flow* as the collection of all packets with the same input and output index values. We further let a *group flow* denote the set of all packets from a given group destined to a unique output. All packets belonging to the same *group flow* will be buffered in the same memory module at their destination output. For example, all packets from group $\#1$ that are destined to output N will be buffered in memory 1 of output N. Hence, multiple memory modules must be maintained at each output to hold packets from different group flows. Clearly, the number of memory modules maintained at each output is G, since for each output, there can be at most G different *group flows*, each of which corresponds to one group. We shall let all packets from the same *flow* traverse through the same path, i.e., we only consider single-path switching, discarding multi-path scenarios that incur packet reordering.

The core switching fabric comprises two stages of passive crosspoint switches. The first connects the ingress ports to the rest of the fabric, hosting a pool of $\alpha \times \alpha K$ crosspoint switches per group, where K denotes the number of crosspoint switches in the second stage and α is the maximal number of outputs that can be matched to a single input. Note that $\alpha = 1$ represents the common case in which each input can be matched to at most one output. If $\alpha > 1$ then a transfer speedup is required. Each of the crosspoint switches in the second stage has $\alpha\frac{N}{G}$ inputs and $\frac{N}{K}$ outputs. By placing an $\alpha \times \alpha K$ switch between the crosspoints pool and each input port, maximal traffic throughput is guaranteed, as will be elaborated on in the following section. From a crosspoint optimization perspective, since the highest number of inputs or outputs on any crosspoint device in the system is a key scalability metric, we note that the maximal port count on any crosspoint is given by $max\{\alpha K, \alpha\frac{N}{G}\}$. For example, if one wishes to design a 512-port switch (i.e. $N = 512$), where $\alpha = 2$, $G = 16$, $K = 8$, then $max\{\alpha K, \alpha\frac{N}{G}\} = 64$,suggesting that the switch can be realized using existing off-the-shelf crosspoint devices.

III. FMWM/OCF Stability Analysis

In this section we derive the necessary conditions for the switch supporting a single class of service to be stable. With reference to figure 1, let $Q_{ij}(t)$ denote the VOQ size at input i holding packets destined to output j at time t. Let us also define the corresponding random arrival process, $A_{ij}(t) \in \{0, 1\}$, with a mean (normalized) rate of packet arrivals from input i to output j, $E[A_{ij}(t)] = \lambda_{ij} \leq 1$. Since the switch is equally partitioned into G independent groups and each group supports its own non-blocking paths, stability analysis focused on any particular group can be easily extended to all other groups with minor modifications, as will be described later. Throughout this paper, we consider a simple FMWM/OCF scheduling algorithm pertaining to the g^{th} group. The algorithm consists of an iterative process whereby during each iteration the maximal weight among the currently contending set of nodes is found, and a match is registered between the corresponding input-output pairs. An iteration example is depicted in figure 2. Upon matching an input to an output, the respective input and output pair is removed from contending during subsequent iterations (shown in scenario 1 of figure 2). Alternatively, only the associated output is removed from future contention, as illustrated in scenario 2 of figure 2, allowing other inputs from the same group to be matched to available outputs. Assuming the weight matrix is not completely null, the number of iterations can range between 1 and N/G.

Configuration of the crosspoints, determined by the FMWM/OCF algorithm, can be represented by a service matrix, $S(t) = \{S_{ij}(t)\}$, where $S_{ij}(t) = 1$ if input i is matched to output j at time t, otherwise $S_{ij}(t) = 0$. Based on the weights of the queues, a schedule is obtained which remains unchanged for k consecutive time slots. A new schedule will only occur at time $t + k$, reflected by $S_{ij}(t + k)$.

Definition 1: Let Λ_g denote the traffic rate matrix for the g^{th} group, as given by

$$\Lambda_g = \begin{bmatrix} \lambda_{\frac{(g-1)N}{G}+1,1} & \lambda_{\frac{(g-1)N}{G}+1,2} & \cdots & \lambda_{\frac{(g-1)N}{G}+1,N} \\ \lambda_{\frac{(g-1)N}{M}+2,1} & \lambda_{\frac{(g-1)N}{M}+2,2} & \cdots & \lambda_{\frac{(g-1)N}{M}+2,N} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_{\frac{gN}{G},1} & \lambda_{\frac{gN}{G},2} & \cdots & \lambda_{\frac{gN}{G},N} \end{bmatrix} \quad (1)$$

Definition 2: Let $\Omega_g(t)$ denote the weight matrix at time t,

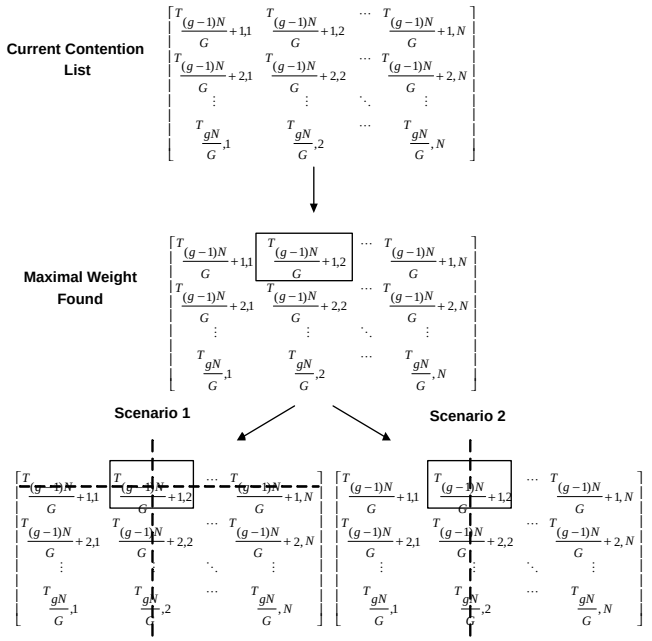

Fig. 2. Example of an iteration at group g, for the FMWM/OCF scheduling algorithm. Each element represents a weighted request for service. Scenario 1 represents a single input/output matching, while Scenario 2 pertains to the case where more than one input from the same group is matched to an output.

such that

$$\Omega_g(t) = \begin{bmatrix} T_{\frac{(g-1)N}{G}+1,1} & T_{\frac{(g-1)N}{G}+1,2} & \cdots & T_{\frac{(g-1)N}{G}+1,N} \\ T_{\frac{(g-1)N}{G}+2,1} & T_{\frac{(g-1)N}{G}+2,2} & \cdots & T_{\frac{(g-1)N}{G}+2,N} \\ \vdots & \vdots & \ddots & \vdots \\ T_{\frac{gN}{G},1} & T_{\frac{gN}{G},2} & \cdots & T_{\frac{gN}{G},N} \end{bmatrix} \tag{2}$$

where $T_{ij}(t)$ is the waiting time of the HoL cell in queue (i,j) at time t.

Definition 3: Let the *queue occupancy vector* for the g^{th} group be defined as

$$Q_g(t) = \left[Q_{\frac{(g-1)N}{G}+1,1}(t), Q_{\frac{(g-1)N}{G}+1,2}(t), ..., Q_{\frac{gN}{G},N}(t) \right]^T. \tag{3}$$

Definition 4: An arrival process is said to be strictly admissible iff $\sum_{i=1}^{N} \lambda_{ij} \leq 1$ and $\sum_{j=1}^{N} \lambda_{ij} \leq 1$.

Definition 5: The (aggregate) weight of the FMWM/OCF algorithm at time t is given by

$$\begin{aligned} W^{FMWM/OCF}(t) &= \sum_{i,j} T_{ij}(t) S_{ij}^{FMWM/OCF}(t) \\ &= \left\langle \Omega_g(t), S_{ij}^{FMWM/OCF}(t) \right\rangle \end{aligned} \tag{4}$$

where $S_{ij}^{FMWM/OCF}(t)$ denotes the matching configurations determined by the scheduling algorithm at time t.

Definition 6: Let $\tau_{ij}^{(m)}(t)$ denote the inter-arrival time between two consecutive cells, m and $m+1$, both of which are stored in queue (i,j), and correspondingly, let $\tau_{ij}(t) = \max\{\tau_{ij}^{(m)}(t),\ m = 1, 2, ..., Q_{ij}(t)\}$.

Definition 7: Let α denote the maximal number of matches allowed to be made for each input port during a single scheduling period.

The buffer dynamics under the FMWM/OCF algorithm dictate that for $Q_{ij}(t) > \eta k$

$$T_{ij}(t+k) = T_{ij}(t) + k - S_{ij}(t) \sum_{m=1}^{\eta k} \tau_{ij}^{(m)}(t), \tag{5}$$

where η is the internal transfer speedup, while for $Q_{ij}(t) \leq \eta k$

$$\begin{aligned} & T_{ij}(t+k) \leq T_{ij}(t) + k \leq Q_{ij}(t) \cdot \tau_{ij}(t) + k \\ \leq\ & \eta k \tau_{ij}(t) + k \end{aligned} \tag{6}$$

from which we can write

$$\begin{aligned} & T_{ij}^2(t+k) - T_{ij}^2(t) \\ \leq\ & k^2 + 2\left[k - S_{ij}(t) \sum_{m=1}^{\eta k} \tau_{ij}^{(m)}(t) \right] T_{ij}(t) \end{aligned} \tag{7}$$

for $Q_{ij}(t) > \eta k$, and

$$\begin{aligned} & T_{ij}^2(t+k) - T_{ij}^2(t) \leq k^2 + 2kT_{ij}(t) \\ \leq\ & k^2 + 2\eta k^2 \tau_{ij}(t) \end{aligned} \tag{8}$$

for $Q_{ij}(t) \leq \eta k$. The term $\eta k S_{ij}(t)$ expresses the ηk consecutive transmissions that may occur during a single frame interval. Next, we construct a discrete-time quadratic Lyapunov function, $L(t)$ [7], such that $L(t) = \sum_{ij} \lambda_{ij} T_{ij}^2(t)$ [8][9][5]. As an expression of a k time slot lag, we write

$$L(t+k) - L(t) = \sum_{ij} \lambda_{ij} \left(T_{ij}^2(t+k) - T_{ij}^2(t) \right). \tag{9}$$

By partitioning the above into the case of $Q_{ij}(t) < \eta k$ and $Q_{ij}(t) \geq \eta k$, we obtain the following:

$$\begin{aligned} & E\left[L(t+k) - L(t) | Q(t) \right] \\ & \leq \sum_{ij} \lambda_{ij} \left(k^2 + 2\left[k - S_{ij}(t) \sum_{m=1}^{\eta k} E\left[\tau_{ij}^{(m)}(t) \right] \right] T_{ij}(t) \right) \cdot \\ & + \sum_{ij} \lambda_{ij} E\left[k^2 + 2\eta k^2 \tau_{ij}(t) \right] \\ & \leq 2 \sum_{ij} \lambda_{ij} \left[k - \frac{\eta k S_{ij}(t)}{\lambda_{ij}} \right] T_{ij}(t) + \sum_{ij} 2(\lambda_{ij} k^2 + \eta k^2) \\ & \leq 2(\eta + \lambda_{ij}) k^2 \frac{N^2}{G} + \sum_{ij} 2T_{ij}(t) \left[k\lambda_{ij} - \eta k S_{ij}(t) \right] \\ & \leq 2k \left[\langle \Lambda_g, \Omega_g(t) \rangle - \eta \left\langle S_g^{FMWM/OCF}(t), \Omega_g(t) \right\rangle \right] + C \end{aligned} \tag{10}$$

where $C = 2(\eta+1)k^2 \frac{N^2}{G}$, is a constant.

In order to prove that the algorithm yields a stable queueing system, we show that beyond a given threshold of maximal weight there is a negative drift in the state of the system. Mathematically speaking, from inequality (10), an appropriate value for η, such that $\langle \Lambda_g, \Omega_g(t) \rangle < \eta \left\langle S_g^{FMWM/OCF}(t), \Omega_g(t) \right\rangle$, guarantees that the algorithm is stable. Hence, we focus

our attention on the two basic scenarios described above, as illustrated in figure 2:

Scenario 1: For each matching generated, its respective input and output pair is removed from contending during subsequent iterations.

Theorem 1: For scenario 1, an $N \times N$ switch running the FMWM/OCF scheduling algorithm with a transfer speedup of 2 is stable under admissible i.i.d traffic for any frame size, k.

Proof: Without loss of generality, assume that following a round of matching, VOQ_{sl}, where $s\epsilon\left\{\frac{(g-1)N}{G}+1,...,\frac{gN}{G}\right\}$ and $l\epsilon[1,...N]$ is selected. It then follows that all of the elements in row s and column l of the weight matrix are removed from future contention. By decomposing $<\Lambda_g,\Omega_g>\leq \eta < S_g,\Omega_g >$ into each round, we have $\sum_{j=1}^{N}\lambda_{sj}+\sum_{i=\frac{(g-1)N}{M}+1}^{\frac{gN}{M}}\lambda_{il}\leq\eta$. For any admissible traffic pattern, we know that $\sum_{j=1}^{N}\lambda_{sj}\leq 1$ and $\sum_{i=\frac{(g-1)N}{G}+1}^{\frac{gN}{G}}\lambda_{il}\leq 1$, from which we deduct that $\sum_{j=1}^{N}\lambda_{sj}+\sum_{i=\frac{(g-1)N}{G}+1}^{\frac{gN}{G}}\lambda_{il}\leq 2$. Hence, $\eta=2$ is sufficient to guarantee stability. ■

Scenario 2: For each matching generated, only the associated output is removed from future contentions. In this case, we remove the restriction of only one VOQ being matched per ingress port, such that there can now be up to α VOQs matched per ingress port.

We extend the stability analysis devised thus far to address scenario 2, i.e. the case in which up to $\alpha > 1$ VOQs can be matched in each ingress port during every *schedule* (note that α is a fixed number, although in each schedule different input ports may have variant number of actual matches, but the number of matches can not exceed α). A schedule here comprises of multiple rounds/iterations, each of which produces one input-output matching. As such, a schedule round may have at most $\frac{\alpha N}{G}$ rounds/iterations per group.

Theorem 2: In the case of scenario 2, an $N \times N$ switch running the FMWM/OCF scheduling algorithm with transfer speedup of $\eta \geqslant 1+1/\alpha$ is stable under admissible i.i.d traffic for any frame size.

Proof: Let us first briefly review the matching process. At the beginning of each schedule interval, the VOQ with largest weight is selected; later all VOQs with the same output as that chosen are removed from subsequent contention rounds. Next, the scheduler chooses the VOQ with the largest weight value among those in the current contention list, and then removes all VOQs with same output as the one chosen. The scheduler also checks to see if the number of matches along the same input has reached α. If so, then it removes all VOQs in the same input from future contention. This process is repeated until no more matchings can be made.

Without loss of generality, assume that a schedule produces a total of β matches in a given interval/frame. Hence, there are $\beta\leq\frac{\alpha N}{G}$ rounds/iterations, each of which corresponds to precisely one match. Further, suppose that during the k^{th} round/iteration, where $k\epsilon\{1,2,...,\beta\}$, VOQ_{i_k,j_k}, which has the largest weight value among those in the current contending list, is selected by the scheduler. Moreover, we let I_k, J_k denote the set of all possible input and output indices for the current contention list, respectively; for example, clearly, if $k=1$, $I_1=\{\frac{(g-1)N}{G}+1,\frac{(g-1)N}{G}+2,...,\frac{gN}{G}\}$ and $J_1=\{1,2,...,N\}$. If the number of matches (including the most recent one) at the same input, e.g. input $i_k\epsilon I_k$, is α, then all *VOQs* in the current contention list, with the same output and input as the selected one, are removed from future rounds.

Let $W_{i_k^1,j_k^1},W_{i_k^2,j_k^2},...,W_{i_k^\alpha,j_k^\alpha}$ denote the weight values of the 1^{st},2^{nd},..., α matching of the input, which were selected by the scheduler during rounds $k_1^{th},k_2^{th},...,k_\alpha^{th}$, respectively. Clearly $W_{i_k^1,j_k^1}\geq W_{i_k^2,j_k^2}\geq\cdots\geq W_{i_k^\alpha,j_k^\alpha}$; hence $W_{i_k^\alpha,j_k^\alpha}\leq\frac{1}{\alpha}(W_{i_k^1,j_k^1}+W_{i_k^2,j_k^2}+...+W_{i_k^\alpha,j_k^\alpha})$. Furthermore, let $I_k^1,...,I_k^\alpha=I_k$ denote the set of all possible input indices for the contending list during rounds $k_1^{th},k_2^{th},...,k_\alpha^{th}$, respectively; and let $J_k^1,...,J_k^\alpha=J_k$ denote the set of all possible output indices for the contending list using similar round notation. We thus have

$$\sum_{j\epsilon J_k}\lambda_{i_k,j}W_{i_k,j}\leq W_{i_k^\alpha,j_k^\alpha}\leq\frac{1}{\alpha}(W_{i_k^1,j_k^1}+W_{i_k^2,j_k^2}+...+W_{i_k^\alpha,j_k^\alpha}), \tag{11}$$

and

$$\sum_{i\epsilon I_k^1}\lambda_{i,j_k^1}W_{i,j_k^1}+\sum_{i\epsilon I_k^2}\lambda_{i,j_k^2}W_{i,j_k^2}+...+\sum_{i\epsilon I_k^\alpha=I_k}\lambda_{i,j_k^\alpha}W_{i,j_k^\alpha}\leq W_{i_k^1,j_k^1}+W_{i_k^2,j_k^2}+...+W_{i_k^\alpha,j_k^\alpha}, \tag{12}$$

hence,

$$\sum_{i\epsilon I_k^1}\lambda_{i,j_k^1}W_{i,j_k^1}+\sum_{i\epsilon I_k^2}\lambda_{i,j_k^2}W_{i,j_k^2}+...+\sum_{i\epsilon I_k^\alpha=I_k}\lambda_{i,j_k^\alpha}W_{i,j_k^\alpha}+\sum_{j\epsilon J_k}\lambda_{i_k,j}W_{i_k,j}\leq(1+\frac{1}{\alpha})(W_{i_k^1,j_k^1}+W_{i_k^2,j_k^2}+...+W_{i_k^\alpha,j_k^\alpha}) \tag{13}$$

Alternatively, if the number of matches (including the new one) at the same input is α, we have

$$\sum_{i\epsilon I_k^1}\lambda_{i,j_k^1}W_{i,j_k^1}+\sum_{i\epsilon I_k^2}\lambda_{i,j_k^2}W_{i,j_k^2}+...+\sum_{i\epsilon I_k^\gamma=I_k}\lambda_{i,j_k^\alpha}W_{i,j_k^\alpha}\leq W_{i_k^1,j_k^1}+W_{i_k^2,j_k^2}+...+W_{i_k^\gamma,j_k^\gamma}<(1+\frac{1}{\alpha})(W_{i_k^1,j_k^1}+W_{i_k^2,j_k^2}+...+W_{i_k^\gamma,j_k^\gamma}) \tag{14}$$

This represents a general result for every round/iteration. Therefore, by decomposing $<\Lambda_g,\Omega_g>\leq\eta<S_g,\Omega_g>$ into each round according to inequalities (13) and (14), we conclude that $<\Lambda_g,\Omega_g>\leq(1+\frac{1}{\alpha})<S_g,\Omega_g>$,which implies that $\eta\geqslant 1+1/\alpha$ is a sufficient condition to guarantee stability. ■

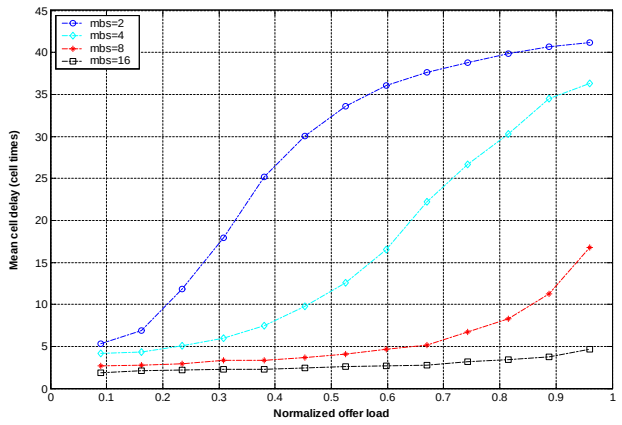

Fig. 3. Average cell delay as a function of the mean burst size (MBS) for a fixed frame size of 8 cells. The FMWM/OCF algorithm can issue at most 1 match per ingress port.

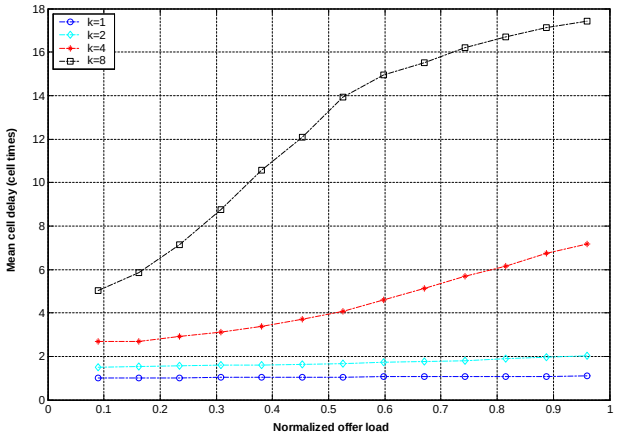

Fig. 4. Average cell delay when arrivals are Bernoulli i.i.d. with uniform distribution for different frame sizes (k). The FMWM algorithm issues up to 2 matches per ingress port.

IV. Simulation Results

In order to evaluate the performance of the FMWM/OCF algorithm under the multi-crosspoints based architecture proposed, three sets of simulations were carried out. In all cases, a 12×12 switch was considered with a transfer speedup of 2. The switch was partitioned into 4 independent switching groups, each of which supported 3 ingress ports. In the first three sets of simulations $\alpha = 1$ (i.e. the transfer speedup is 2).

The first set of simulations was targeted at examining the impact of bursty traffic on the delay characteristics. A two-state Markov-modulated (ON/OFF) process was employed [10], whereby bursts are uniformly distributed across the outputs. Figure 3 shows the average delay as a function of the mean burst sizes (MBS) for a fixed frame size of 8 packets. An inverse relationship between the MBS and the average delay is observed. Since the FMWM scheduling discipline is inherently correlated, bursty traffic better utilizes the transmission intervals.

In the second set of simulations, the FMWM/OCF algorithm was allowed to make up to 2 matches per ingress port and the transfer speedup is correspondingly dropped to 1.5. The arrival process was Bernoulli i.i.d with uniformly distributed destination distribution. Figure 4 depicts the average delay measured for different frame sizes and shows that despite the relaxed switching times and distributed passive crosspoint switches, the overall performance is kept high.

V. Conclusions

This paper presents a novel scalable multi-crosspoints based packet switching architecture coupled with a frame-based scheduling algorithm for routers with large port densities and high-speed line rates. It has been shown that the architecture can guarantee 100% throughput for a broad class of traffic scenarios. By equally partitioning an $N \times N$ CIOQ switch into multiple independent switching groups, the timing requirements from the FMWM/OCF algorithm are substantially reduced. Moreover, by reconfiguring the crosspoint switches once every several time slots, it is possible to significantly relax the timing constraints imposed on the scheduling algorithm. Compared with other architectures targeting high end routers, the proposed multi-crosspoints based architecture is scalable, easy to implement, and does not entail complex packet processing or reordering.

Acknowledgements

This work has been partially supported by the Department of Energy research contract DE-FG02-04ER25607.

References

[1] R.Z.S.Iyer and N.Mckeown, "Analysis of the parallel packet switch architecture," *IEEE/ACM Transactions on Networking*, vol. 11, no. 2, pp. 314–324, 2003.

[2] R. Iyer and N. Mckeown, "Routers with a single stage of buffering," *ACM Computer Communication Review SIGCOMM '02*, pp. 251–264, 2002.

[3] C. Chang, D. Lee, and Y. Jou, "Load balanced Birkhoff-von Neumann switches," *High Performance Switching and Routing, 2001 IEEE Workshop*, pp. 276–280, 2001.

[4] I.Keslassy, "The load-balanced router," *Ph.D. Thesis, Stanford University*, 2004.

[5] X.Li and I.Elhanany, "Stability of a frame-based maximal weight matching algorithm with transfer speedup," *IEEE Communications Letters*, vol. 9, no. 10, pp. 942–944, 2005.

[6] S.Y.Nam and D.K.Sung, "Decomposed corssbar switches with multiple input and output buffers," *IEEE GLOBECOM 2001*, vol. 4, pp. 2661–2665, 2001.

[7] J. Dai and B. Prabhakar, "The throghput of data switches with and without speedup," *IEEE INFOCOM 2000*, pp. 556–564, March 2000.

[8] A. Mekkittikul and N. McKeown, "A practical scheduling algorithm to achieve 100% throughput ininput-queued switches," in *INFOCOM '98.*, vol. 2, San Francisco, CA, USA, Mar./Apr. 1998, pp. 792–799.

[9] P. Kumar and S. Meyn, "Stability of queueing networks and scheduling policies," *IEEE Transactions on Automatic Control*, vol. 40, no. 2, pp. 251–260, February 1995.

[10] I. Elhanany and B. Matthews, "On the performance of output queued cell switches with non-uniformly distributed bursty arrivals," *IEE Proceedings on Communications*, vol. 153, no. 2, pp. 201–204, 2006.