

# A Dual-Level Matching Algorithm for 3-Stage Clos-Network Packet Switches

H. Jonathan Chao, Soung Y. Liew, and Zhigang Jing  
 Polytechnic University, Brooklyn, New York 11201  
 chao@poly.edu, syliew@queens.poly.edu, zgjing@kings.poly.edu

## Abstract

In this paper, we present a new dual-level matching algorithm for 3-stage Clos-network packet switches, called **d-MAC**. Using a two-level matching algorithm, namely module-level matching and port-level matching, **d-MAC** is highly scalable and maintains high system performance. The module-level matching is responsible for finding the module-to-module matching according to the queue status of the switch, while the port-level matching is responsible for determining port-to-port matching and route assignment simultaneously. The two-level matchings are computed in a pipelined and parallel manner to speedup packet scheduling.

## 1. Introduction

Current router technologies cannot provide switching capacity large enough to meet current and future Internet bandwidth demands. Because the number of switching elements of single-stage switches is proportional to the square of the number of switch ports, they are unattractive as the switch size becomes large. On the other hand, the multi-stage switch architecture, such as the 3-stage Clos-network, is able to provide much higher switch capacity, e.g., up to a few hundred terabit/s or even petabit/s [1].

To schedule cells from input ports to output ports in the bufferless 3-stage Clos-network switch, there are two major issues: cell scheduling and route assignment. The former is necessary for determining a set of port-to-port matches for cell switching, while the latter is necessary for assigning a set of internally conflict-free paths for the above matches through the 3-stage Clos-network switch.

It is very challenging to find an efficient and fast scheduling scheme to provide high throughput, starvation-free, acceptable delay, and fairness performance under various traffic conditions for a bufferless 3-stage Clos-network switch architecture. In [2], a path-switching scheme was proposed for scheduling in the bufferless 3-stage Clos-network packet switch, where a periodic IM-to-OM route assignment is predetermined prior to cell arrivals in accordance with a given traffic-loading pattern. With this predetermined assignment, cells can be transferred when there are routes

available for the corresponding IM-OM pairs. However, the path-switching scheme assumes the traffic-loading pattern is known in advance. This is not applicable to the Internet because most Internet traffic is connectionless. In [3], authors proposed a distributed static round-robin (Distro) scheduling algorithm to route cells in a round-robin manner. However, the Distro algorithm cannot handle various traffic conditions well due to its static nature. On the other hand, our previous proposed frame-based matching algorithm for Clos-network switches, called **f-MAC**, has a more general assumption on the traffic loading to the switch [4]. To compensate for the arbitration time needed for determining port-to-port matching and routing assignments, **f-MAC** packs  $r$  cells into a frame as a transmission unit. However, the time complexity of **f-MAC** is quite high, which prevents one from selecting a small frame size.

## 2. Switch architecture

Figure 1 depicts the proposed switch architecture for **d-MAC**. There are two major components: switch fabric and packet scheduler (PS). The switch fabric is responsible for routing cells, while the PS is responsible for determining routing paths of transferring cells in the switch fabric.

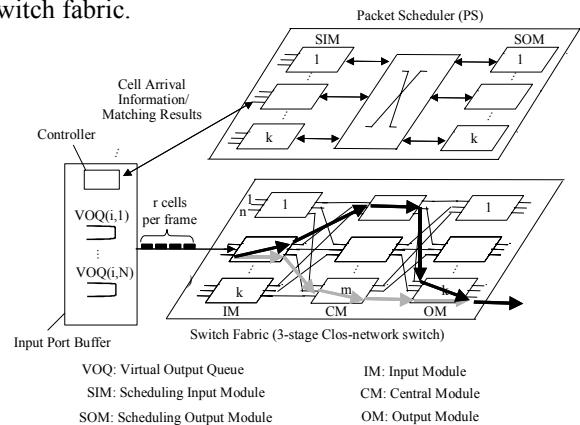


Figure 1. The switch architecture.

The switch fabric is the bufferless 3-stage Clos network, in which  $k$  input modules (IMs) and  $k$  output modules (OMs) are in the first and third stages,

respectively, and  $m$  central modules (CMs) in the middle stage. Each IM (OM) has  $n$  input (output) ports, where  $nk = N$  is the network size. In addition, each input port  $g$  maintains  $N$  virtual output queues (VOQs), in which  $VOQ(g,h)$  is dedicated for output port  $h$ .

The PS consists of  $k$  scheduling input modules (SIMs) and  $k$  scheduling output modules (SOMs), each of which corresponds to an input switch module (or output switch module) in the Clos-network switch. SIMs and SOMs can communicate via a crosspoint switch in the center of the PS so that they can gather necessary information to compute the port-to-port matching and routing assignments in a distributed manner.

With the increases of switch size and port speed, the scheduling time for resolving port-to-port matching and route assignment becomes more stringent. To relax the strict arbitration time constraint, d-MAC operates based on a frame of  $r$  cells ( $r > 1$ ). In each frame, the switch fabric maintains the same switching state so that up to  $r$  cells can be transferred from a particular input port to a particular output port. The switching state is allowed to change from frame to frame, nevertheless. The frame size is pre-selected depending on such things as assignment arbitration time, round-trip delay between the input ports to the PS, and reconfiguration time of switch fabric.

### 3. d-MAC

#### 3.1. Overview

To determine the switching state of the switch fabric, the PS does two levels of assignments: module-level matching and port-level matching. With reference to Figure 2, the module-level matching determines  $k'$  ( $k' < k$ ) sets of SIM-SOM matching for each frame such that only ports on the corresponding IM-OM pairs can be matched in the port-level matching.

For the module-level matching, the switching patterns of a number of, say  $F$ , frames can be determined simultaneously. These  $F$  frames constitute a super-frame. We use the example shown in Figure 2 to illustrate the module-level matching steps of d-MAC as follows. With reference to Figure 2(a), a traffic matrix is used to represent the queue status of the switch, in which entry  $(i, j)$  denotes the number of buffered cells that desire to be transmitted from  $IM_i$  to  $OM_j$ . According to this traffic matrix, a request matrix can be obtained. Note that the request matrix gives the number of requests that can be sent from each SIM to each SOM. Then a scheduling algorithm can be employed to do arbitration among these requests, and this process produces a super-frame matrix, in which each entry represents the matching opportunities between each SIM and each SOM in one super-frame. With reference to Figure 2(b), the super-frame matrix is

further decomposed into the module-level matching matrices, where a module-level matching matrix represents the matching pattern of SIMs and SOMs in one matching cycle as shown in Figure 2(c). The module-level matching is done after the module-level matching matrices are determined.

With each of these module-level matching matrices, the port-level matching can thereafter perform the task of matching cycle, that is, port-to-port matching and route assignment, for the given SIM-SOM pairs. For example, ports on IM1 can only be matched with ports on either OM2 or OM3. A possible port-level matching assignment for IM1 is shown in Figure 3. Note that each module-matching matrix is a  $k \times k$  matrix of which the row/column sum is either zero or one (unmatched) or one (matched).

$$\begin{matrix} \begin{pmatrix} 16 & 1 & 11 & 2 \\ 2 & 12 & 0 & 4 \\ 7 & 10 & 11 & 5 \\ 2 & 6 & 7 & 8 \end{pmatrix} & \rightarrow & \begin{pmatrix} 4 & 1 & 3 & 1 \\ 1 & 3 & 0 & 2 \\ 2 & 3 & 3 & 2 \\ 1 & 2 & 2 & 2 \end{pmatrix} & \rightarrow & \begin{pmatrix} 2 & 1 & 2 & 1 \\ 1 & 3 & 0 & 2 \\ 1 & 1 & 3 & 1 \\ 1 & 1 & 1 & 2 \end{pmatrix} \\ \text{Traffic Matrix} & & \text{Request Matrix} & & \text{Super-frame Matrix} \end{matrix}$$

(a) Determining Super-frame Matrix

$k'$  matching cycles in one frame

$$\begin{pmatrix} 2 & 1 & 2 & 1 \\ 1 & 3 & 0 & 2 \\ 1 & 1 & 3 & 1 \\ 1 & 1 & 1 & 2 \end{pmatrix} \rightarrow \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} + \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix} + \dots + \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

(b) Super-frame Matrix Decomposition

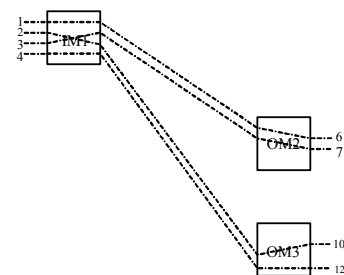
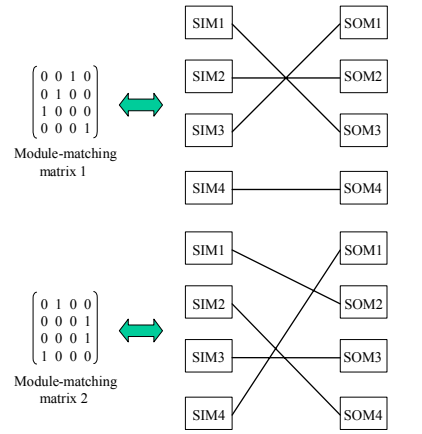


Figure 2. Module-level matching of d-MAC.

Figure 3. Port-level matching of d-MAC.

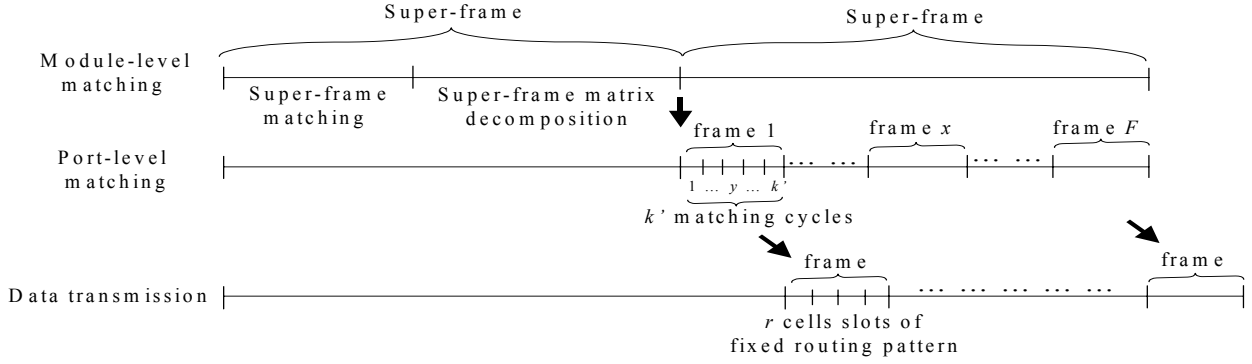


Figure 4. Pipelined process in the dual-level matching scheme.

The module-level matching and port-level matching assignments are computed in the PS in a pipelined manner. With reference to Figure 4, to find the module-level matching, time is divided into super-frames, each having  $F$  frames. In each super-frame,  $F$  sets of  $k'$  module-matching matrices for frames in the next super-frame are determined. The module-level matching algorithm is composed of two phases, namely super-frame matching and super-frame decomposition. To find the port-level matching, each frame is further divided into  $k'$  matching cycles. In each matching cycle, port-to-port matches as well as the route assignment are determined for the IM-OM pairs that correspond to a particular module-matching matrix. After port-level matching is done, cells can then be transmitted according to the assignment that lasts for  $r$  cell slots.

### 3.2. Module-level matching

The module-level matching algorithm is composed of two phases, i.e., super-frame matching and super-frame decomposition.

**3.2.1 Super-frame matching.** The super-frame matching is to determine a super-frame matrix in accordance with the queue status of the switch. Each entry in the super-frame matrix represents the matching opportunities between each SIM and each SOM in one super-frame. With reference to Figure 5, a super-frame matrix is a  $k \times k$  matrix with

- (i) no row/column sum greater than  $F \times k'$ , and
- (ii) no entry greater than  $F$ .

Condition (ii) is stated so because each

$$\begin{matrix} \leq F \\ \leq F \times k' \end{matrix} \begin{bmatrix} 2 & 1 & 2 & 1 \\ 1 & 1 & 3 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{k' \text{ module-matching matrices that belong to the same frame}} + \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}}_{F \times k' \text{ module-matching matrices}} + \dots + \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

SIM- pair needs to be paired up at most once in a frame, thus at most  $F$  times in a super-frame. SOM

Figure 5. The super-frame matrix with  $F=3, k'=2, k=4$ .

We modify the iterative matching algorithm (e.g. iSLIP [5]) to find the super-frame matrix. In the iterative super-frame matching algorithm, the number of requests that can be sent from SIM $i$  to SOM $j$  can be determined as follows. Let  $C(g, h)$  be the number of cells in VOQ( $g, h$ ),  $Q_{i,j} = \sum_{g=n(i-1)+1}^{ni} \sum_{h=n(j-1)+1}^{nj} C(g, h)$  the total length of all VOQs from IM $i$  to OM $j$ . Assume that  $X_{i,j}$  is the number of requests from SIM $i$  to SOM $j$ . A threshold value,  $Q$ , for VOQs is pre-selected (e.g.  $Q=n \times r$ , that is the maximum number of cells which could be sent from one IM to one OM in one frame time.) such that  $X_{i,j} = \min\{\lceil Q_{i,j}/Q \rceil, F\}$ . For example, if  $Q=80$  (e.g.,  $n=8, r=10$ ),  $Q_{i,j}=450$  and  $F=3$ ,  $X_{i,j} = \min\{\lceil 450/80 \rceil, 3\} = \min\{6, 3\} = 3$ . To prevent the starvation problem between IMs and OMs, we take the ceiling function (i.e.,  $\lceil \cdot \rceil$ ) in above operations. In this way, an SIM will send out at least one request whenever there are cells going from an IM to an OM.

In the beginning of each iteration, let  $M_{i,j}$  be the number of matches found for the SIM $i$ -SOM $j$  pair. Assume that  $M_i = \sum_{j=1}^k M_{i,j}$ , and  $O_j = \sum_{i=1}^k M_{i,j}$ . The iterative super-frame matching consists of three steps as follows.

- (i) *Request*: SIM $i$  sends  $R_{i,j}$  requests to SOM $j$ , where  $R_{i,j} = \min\{X_{i,j} - M_{i,j}, F \times k' - M_i\}$ .
- (ii) *Grant*: Each SOM grants up to  $F \times k' - O_j$  requests.
- (iii) *Accept*: Each SIM accepts up to  $F \times k' - M_i$  grants.

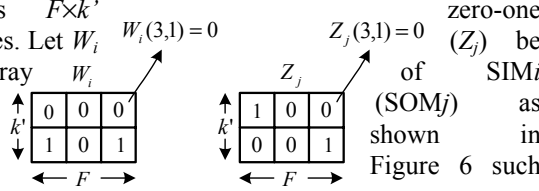
For example, suppose that  $X_{i,j}=3, F=3$  and  $k'=2$ . In the first iteration,  $M_{i,j}=0$ , thus SIM $i$  sends  $R_{i,j} = \min\{X_{i,j}, F \times k'\} = \min\{3, 6\} = 3$  requests to SOM $j$  in step (i). Note that, we can have different schemes, such as the frame-based matching algorithm [6], to resolve the contentions in (ii) and (iii). We skip the related discussion because it is out of the scope of this paper. In this paper, we assume that

each SOM cyclically grants the requests by choosing one request from each SIM in turn. The pointers of SIM and SOM arbiters are updated to one position beyond the accepted position if and only if the grant is accepted in (iii). For example, suppose that only two of the three requests from SIM<sub>*i*</sub> to SOM<sub>*j*</sub> were successfully granted and accepted in the first iteration, then  $M_{i,j}=2$  at the beginning of the second iteration. After several iterations, we can obtain the super-frame matrix with entries  $M_{i,j}$ .

**3.2.2 Super-frame matrix decomposition.** The purpose of super-frame matching decomposition is to decompose the super-frame matrix into  $F \times k'$  module-matching matrices. Each module-matching matrix records the matching status between the SIMs and SOMs in one matching cycle of the next super-frame.

This kind of matrix decomposition problems can be solved by an edge-coloring algorithm. Optimal edge-coloring algorithms are not preferable here, because they either have a high time complexity [7] or are difficult to implement [8]. Instead, we use the parallel matching heuristic, which is modified from Karol's algorithm [9], for decomposing the super-frame matrix. This algorithm contains  $k$  rounds. In each round, an SIM can communicate with one of  $k$  SOMs. Initially, all module-matching matrices have all zero entries, and they are updated as follows.

Each SIM/SOM maintains a two-tuple array that contains  $F \times k'$  zero-one variables. Let  $W_i$  ( $W_i(3,1)=0$ ) be the array of SIM<sub>*i*</sub> and  $Z_j$  ( $Z_j(3,1)=0$ ) be the array of SOM<sub>*j*</sub> as shown in Figure 6 such that,



$$W_i(x, y) = \begin{cases} 0, & \text{if SIM } i \text{ is unmatched in cycle } y \text{ of frame } x, \\ 1, & \text{if it has been matched in cycle } y \text{ of frame } x; \end{cases}$$

$$Z_j(x, y) = \begin{cases} 0, & \text{if SOM } j \text{ is unmatched in cycle } y \text{ of frame } x, \\ 1, & \text{if it has been matched in cycle } y \text{ of frame } x; \end{cases}$$

where  $1 \leq x \leq F$  and  $1 \leq y \leq k'$ .

**Figure 6. Arrays for parallel matching.**

When SIM<sub>*i*</sub> is communicating with SOM<sub>*j*</sub>, d-MAC tries to find as many common zero entries in  $W_i$  and  $Z_j$  as possible to meet the number given by entry  $(i, j)$  in the super-frame matrix. Note that, no more than one zero entries in the same frame can be assigned to the same

SIM-SOM pair. This is because each SIM-SOM pair doesn't need to match more than once in the same frame period. For example, suppose that  $F=3$ , entry  $(i, j)$  equals 3 in the super-frame matrix, and arrays  $W_i$  and  $Z_j$  are as shown in Figure 7. There are three common zero entries of  $W_i$  and  $Z_j$ , that is  $W_i(2,1)=Z_j(2,1)=0$ ,  $W_i(2,2)=Z_j(2,2)=0$ , and  $W_i(3,1)=Z_j(3,1)=0$ . However, since  $(2,1)$  and  $(2,2)$  belong to the same frame (i.e., frame 2), so at most one of them can be assigned to SIM<sub>*i*</sub> and SOM<sub>*j*</sub>. Thus, we can only assign two matching cycles, which are  $(2,1)$  and  $(3,1)$ , for SIM<sub>*i*</sub> and SOM<sub>*j*</sub>.

In addition to updating  $W_i$  and  $Z_j$ , after each round of parallel matching, d-MAC records the decomposition results in the corresponding module-matching matrices. For the above example, module-matching matrices for cycles  $(2,1)$  and  $(3,1)$ , are updated, respectively, as shown in Figure 7.

**Figure 7. Updating matrices of cycles (2,1) and (3,1).**

**3.3. Port-level matching**

When the module-level matching is completed, the matching sequence between SIMs and SOMs (recorded in the module-matching matrices) is determined for the next super-frame. The port-level matching algorithm consists of  $k'$  cycles in a frame, each corresponds to a module-matching matrix. In each matching cycle, the port-level matching algorithm includes two steps:

- (i) port-to-port matching assignment;
- (ii) route assignment.

**3.3.1. Port-to-port matching assignment.** To find the port-to-port matching for the corresponding pair of IM-OM, we use the iterative request/grant/accept algorithm, e.g., iSLIP. For instance, if SIM<sub>*i*</sub> matches with SOM<sub>*j*</sub> in one matching cycle according to the module-matching matrix, SOM<sub>*j*</sub> first sends port availability information to SIM<sub>*i*</sub> at the beginning of this matching cycle. Then, SIM<sub>*i*</sub> performs iterative matching (e.g., iSLIP) locally.

In order to improve the matching efficiency and bandwidth utilization in the frame-based matching schemes, we introduce high-priority and low-priority arbiters in the SIMs. In such a way that VOQs with queue length of more than  $r$  cells will send out high-priority requests and have higher priority to be matched than the unfilled ones. On the other hand, currently matched input

and output pairs may not be able to maintain matching if their corresponding queue lengths are less than  $r$ .

To avoid some pathological traffic conditions that may lead to starvation, the head-of-line (HOL) frame's priority level is boosted to the highest when its waiting time exceeds a certain threshold, say  $Th_w$ . When the HOL frame's waiting time exceeds  $Th_w$ , its request will be set to the highest-priority.

**3.3.2. Route assignment.** In addition to the port-to-port matching assignment, we must also determine the route assignment for cell routing in the switch. The route assignment is performed right after the port-to-port matching assignment, by a heuristic parallel matching scheme modified from Karol's approach [9].

To do this, each SIM/SOM maintains a one-tuple array that contains  $m$  zero-one variables, where  $m$  is the number of CMs in the Clos network. Let  $A_i$  ( $B_j$ ) be such array of SIM $i$  (SOM $j$ ) such that,

$$A_i(z) = \begin{cases} 0, & \text{if CM } z \text{ is not used by IM } i, \\ 1, & \text{if CM } z \text{ has been used by IM } i, \end{cases}$$

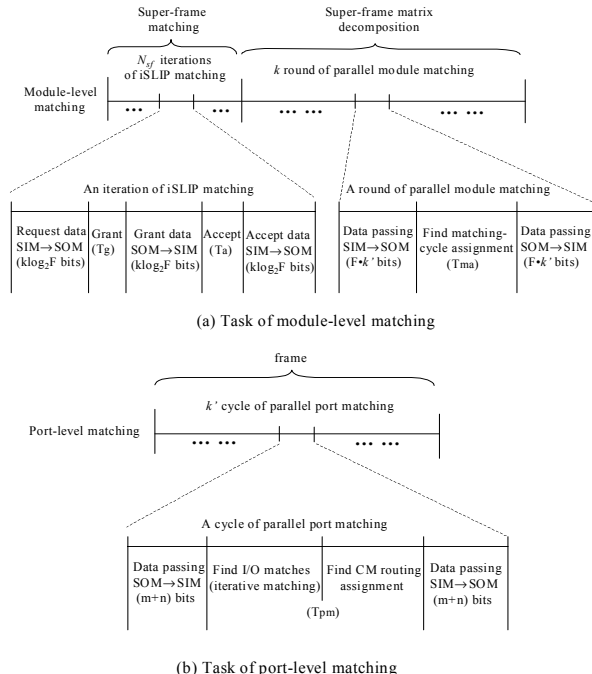
$$B_j(z) = \begin{cases} 0, & \text{if CM } z \text{ is not used by OM } j, \\ 1, & \text{if CM } z \text{ has been used by OM } j, \end{cases}$$

where  $1 \leq z \leq m$ .

In the beginning of each matching cycle, in addition to port availability information, SOM $j$  also sends array  $B_j$  to SIM $i$ , if SIM $i$  and SOM $j$  are matched in this matching cycle. After the port-to-port matching assignment has been determined, d-MAC tries to find as many common zero entries in  $A_i$  and  $B_j$  as possible to meet the number of port-to-port matches for SIM $i$ -SOM $j$  pair. Note that, similar to the port-to-port matching assignment, the route assignment is performed locally in SIM $i$ .

## 4. Time complexity

In this section, we estimate the time complexity of d-



MAC. d-MAC includes two-level matching algorithms, namely module-level matching and port-level matching. Figures 8(a) and 8(b) show the tasks of module-level matching and port-level matching, respectively.

**Figure 8. Tasks of dual-level matching algorithm.**

With reference to Figure 8(a), in super-frame matching phase, each SIM sends at most  $F$  requests to each SOM in each iteration. Hence, there are totally  $k \cdot \log_2 F$  bits request information needed to be passed on from an SIM to all  $k$  SOMs. The amount of time needed to pass the request information is  $k \cdot \log_2 F / BW$ , where  $BW$  is the I/O bandwidth of the crosspoint switch in the packet scheduler as shown in Figure 1. Assume that it takes  $T_g$  for each SOM to grant these requests. Thereafter, each SOM needs to send  $\log_2 F$  bits of data back to each of the  $k$  SIMs. Assume that it takes  $T_a$  for each SIM to accept these grants. Finally, each SIM needs to send  $\log_2 F$  bits of accept acknowledgement to each of the  $k$  SOMs.

Let us assume  $T_{sf} = T_g + T_a$ . Suppose that  $N_{sf}$  iterations of super-frame matching are needed in order to guarantee convergence of the algorithm. The amount of time needed for the super-frame matching is:

$$N_{sf} \cdot \left[ \frac{3 \cdot k \cdot \log_2 F}{BW} + T_{sf} \right]$$

On the other hand, in super-frame matrix decomposition phase,  $k$  rounds of parallel matching are needed to find module-matching matrices. We further assume that it takes  $T_{ma}$  for each SOM to find the module-matching-matrix assignment. Then we have the amount of time needed for the super-frame decomposition be

$$k \cdot \left[ \frac{2 \cdot F \cdot k'}{BW} + T_{ma} \right]$$

Overall, the constraint for the module-level matching is

$$F \cdot r \cdot T_{cell} \geq N_{sf} \cdot \left[ \frac{3 \cdot k \cdot \log_2 F}{BW} + T_{sf} \right] + k \cdot \left[ \frac{2 \cdot F \cdot k'}{BW} + T_{ma} \right]$$

$$\Rightarrow r \cdot T_{cell} \geq \frac{2 \cdot k \cdot k'}{BW} + \left\{ \frac{N_{sf} \cdot \left[ \frac{3 \cdot k \cdot \log_2 F}{BW} + T_{sf} \right] + k \cdot T_{ma}}{F} \right\} \quad (1)$$

With reference to Figure 8(b), in a matching cycle of port-level matching, each SOM sends  $(m+n)$  bits to the corresponding SIM. These bits indicate the availabilities of CMs and output ports of the OM to the IM. Assume it takes  $T_{pm}$  for each SIM to complete (i) iterative port-to-port matching using iSLIP, and (ii) the CM assignment. Before the matching cycle is ended, each SIM sends  $(m+n)$  bits back to the corresponding SOM as an assignment acknowledgement. The time constraint of port-level matching is given by

$$r \cdot T_{\text{cell}} \geq k' \cdot \left[ \frac{2 \cdot (m+n)}{BW} + T_{\text{pm}} \right] \quad (2)$$

A numerical example is given as follows. Suppose that  $n = k = m = 128$ ,  $BW = 40\text{Gb/s}$ ,  $k' = 8$ ,  $F = 16$ ,  $N_{\text{sr}} = \log_2 k = 7$ ,  $T_{\text{sm}} = T_{\text{ma}} = T_{\text{arb}} = 10\text{ns}$ . Constraints of Eqs. (1) and (2) give 152.4 and 182.4 ns, respectively. Thus, the time constraint of d-MAC is 182.4 ns, and  $F$  should be set to 4 ( $r=4$ ) cell time slots for 10-Gbit/s port speed (the cell time slot is equal to 51.2ns at 10-Gbit/s port speed with cell size of 64 bytes). Note that the switch capacity of the crosspoint switch in the PS is only 1/30 of that of the 3-stage Clos-network switch (i.e., 5.12 Tbit/s/160 Terabit/s).

## 5. Performance study

Figure 9 shows the average cell delay of the d-MAC scheme under uniform traffic with different super-frame sizes ( $F$ ) and numbers of matching cycles ( $k'$ ) for a switch size of 64 ( $n=k=8$ ). We assume the frame size ( $r$ ) to be 10 cells, and the CM number ( $m$ ) to be 15. The number of iterations ( $N_{\text{sr}}$ ) to find the super-frame matrix is 4, and that to perform port-to-port matching (i.e., iSLIP) in each matching cycle ( $itr$ ) is 1. The traffic is assumed to be bursty with an average burst length  $l$  of 16 cells. The unbalanced load ( $w$ ) is defined as each input sends a portion of traffic to a particular output port with the remaining evenly distributed to all others. As shown in

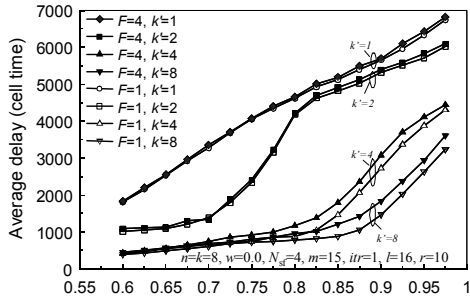


Figure 9, the average delay is not

sensitive to  $F$ , but very sensitive to  $k'$ . The larger  $k'$  is, the more module-to-module matching combination we can have in a frame, hence the more port-to-port matches can be found, which leads to the better delay performance. However, the smaller  $k'$  can relax the time complexity so that the scheme is more implementable with the existing hardware technologies. This allows d-MAC to provide flexibility to tradeoff timing complexity with delay performance.

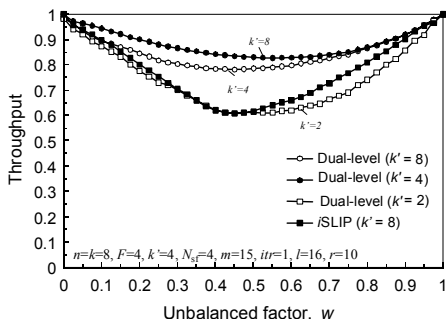


Figure 9. Average cell delay (uniform traffic).  
Figure 10. Throughput under unbalanced traffic.

Figure 10 shows the throughput of d-MAC with different number of matching cycles  $k'$  in one frame under the unbalanced traffic. With  $F=4$ , the throughput degrades when the traffic distribution is unbalanced. However, the throughput is improved with more matching cycles in a frame, e.g.,  $k' \geq 4$ . Because the dual-level matching scheme utilizes the queue length information when performing the module-level matching, the modules with longer queue length are given more matching opportunities in each super-frame. Thus, the dual-level matching scheme with  $k' \geq 4$  can achieve higher throughput than the iSLIP scheme even under unbalanced traffic distribution.

## 6. Conclusion

To scale packet switches to a few hundred terabit/s or even petabit/s, we have developed an efficient packet-scheduling scheme, called the dual-level matching scheme for Clos-network switches (d-MAC), to resolve the scheduling problems in the bufferless 3-stage Clos-network switch. This algorithm consists of two levels of matching, namely module-level matching and port-level matching. Based on the new pipeline-based two-level matching algorithms, d-MAC is highly scalable while maintaining high system performance.

## 7. References

- [1] H. J. Chao, "Next Generation Routers," invited paper, *Proceedings of the IEEE*, vol. 90, no. 9, Sep. 2002, pp. 1518-1558.
- [2] T. T. Lee and C. H. Lam, "Path switching - A quasi-static routing scheme for large-scale ATM packet switches," *IEEE J. Select. Areas Commun.*, vol. 15, pp. 914-924, June 1997.
- [3] K. Pun and M. Hamdi, "Distro: A Distributed Static Round-Robin Scheduling Algorithm for Bufferless Clos-Network Switches," *IEEE GLOBECOM'2002*, Taipei, Nov 17-21, 2002.
- [4] H. J. Chao, K-L. Deng, and Z. Jing, "A Petabit Photonic Packet Switch (P<sup>3</sup>S)," *IEEE INFOCOM'2003*, San Francisco, April 1-3, 2003.
- [5] N. McKeown, "The iSLIP scheduling algorithm for input-queues switches," *IEEE/ACM Trans. On Networking*, pp. 188-200, April, 1999.

- [6] A. Bianco, *et al.*, "Frame-based matching algorithms for input-queued switches," *Workshop on High Performance Switching and Routing*, Kobe, Japan, 26-29 May 2002
- [7] R. Cole, K. Ost, S. Schirra, "Edge-coloring bipartite multigraphs in  $O(E \log D)$  time", *Combinatorica 21 (2001)*, pp 5-12.
- [8] T. T. Lee and S. Y. Liew, "Parallel routing algorithms in Benes-Clos network," *IEEE Trans. on Commun.*, vol. 50, no. 11, Nov. 2002, pp. 1841-1847.
- [9] M. Karol, and C-L. I, "Performance Analysis of a Growable Architecture for Broadband Packet (ATM) Switching," *IEEE Trans. on Commun.*, vol. 40, no. 2, pp 431-439, Feb. 1992.