Scalable Two-Stage Clos-Network Switch and Module-First Matching

Roberto Rojas-Cessa and Chuan-Bi Lin

***Abstract*— Clos-network switches were proposed as a scalable architecture for the implementation of large-capacity circuit switches. In packet switching, the three-stage Clos-network architecture uses small switches as modules to assemble a switch with large number of ports or aggregated ports with high data rates. However, the configuration complexity of packet Clos-network switches is high as port matching and path routing must be performed. In the majority of the existing schemes, the configuration process performs routing after port-matching is achieved, and thus making port matching expensive in hardware and time complexity for a large number of ports. Here, we reduce the configuration complexity by performing routing first and port matching afterwards in a three-stage Clos-network switch. This approach applies the reduction concept of Clos networks to the matching process. This approach results in a feasible size of schedulers for up to Exabit-capacity switches, an independent configuration of the middle stage modules from port matches, a reduction of the matching communication overhead between different stages, and a release of the switching function to the last-stage modules in a three-stage switch. By using this novel matching scheme, we show that the number of stages of a Clos-network switch can be reduced to two, and we call this the two-stage Clos-network packet switch.**

***Index Terms*— Clos network, two-stage switch, input queued, module matching, Exabit switch.**

I. Introduction

Clos-network switches were proposed as a scalable architecture for the implementation of large-capacity circuit switches [1]. The three-stage Clos-network architecture uses small switches as modules in each stage to assemble a large-size switch. Each of this modules can use a crossbar fabric. Then, the total number of crosspoints and switch modules can be engineered for economy. Input-queued (IQ) Clos-networks have queues in the input ports to store cells when there is input and output contention. However, the configuration of Clos-network packet switches is complex as output contention and path routing need to be resolved before the transmission of packets takes place.

In this paper, we follow the practice of segmenting variable length packets into fixed-size packet, named cells (not necessarily those of Asynchronous Transfer Mode) at their arrival, and re-assembling variable-length packets at their departure from the switch.

Although Clos-network switches reduce the hardware amount, in terms of the number of crosspoints, the module size, and the number of modules required to implement high-capacity packet switches, there are other issues that can limit their scalability: a) The time for configuring all modules before a packet is sent through the switch. This requires a fast packet scheduler and an efficient exchange of scheduling information among the arbiters that select the packets that are to be switched at a given time slot. b) The number of matching units, (e.g., ports) as a large number of matching ports would require large arbiters implemented in hardware. For a switch with large number of ports N, a matching process would involve all N ports. The implications of this is high-hardware complexity that may not allow a centralized implementation of schedulers, and long resolution times. As an example, a scheduler for a switch with $N = 1024$ would be difficult to build. To the best of our knowledge, implementation of schedulers of up to $N \leq 64$ have been reported as practical for implementation [2]. c) Time relaxation for the configuration of all modules, or an asynchronous configuration, as each input port may be located far from each other. The time to configure the switch not only implies fast arbiters as in a), but also that the exchange of control information must travel the smallest inter-chip distances to decrease the delay overhead. For example, for a distributed implementation of a scheduler, which may be segmented into parts placed in different modules, the scheduling process may need to exchange requests and grants between different modules and these may be far from each other.

One strategy to simplify the configuration complexity of Clos-network switches is to use buffers in the modules of the first and third stages of a three-stage switch [3]. In this way, the scheduling of packets becomes a dispatching scheme issue. Several dispatching schemes have been proposed to achieve high performance while keeping the transmission of cells in sequence [3]-[6]. However, the buffers in the first-stage modules need to work with a speedup of $n+1$ and those in the third-stage modules need to work with a speedup of $m+1$, where n is the number of input ports of the module switches in the first stage and the number of outputs of the module switches of the third stage, and m is the number of switch modules in the second stage. Therefore, we consider that IQ Clos-network switches, with bufferless switch modules, are a good alternative for large scale switches.

Considering that IQ Clos-network switches have no memory in any stage, but in the input ports, the switch modules are simple to design. An IQ Clos-network switch needs no memory speedup in the switch modules and is free of out-of-sequence forwarding that may occur in buffered Clos-network switches. As the input ports have virtual output queues (VOQs), where one queue per output port is allocated to store cells for that output, the IQ Clos-network switch avoids the head-of-line (HOL) blocking problem [7]. A variety of matching schemes to configure IQ Clos-network switches have been proposed [8]-[11]. These schemes show how to achieve high performance under uniform traffic. Several schemes solve the configuration process in two phases: port matching in the first phase and routing afterwards, as the routing process uses the results of the port match. The matching schemes based on port matching can be complex and time consuming. For example,

This work is supported in part by National Science Foundation under Grant Awards 0435250 and 0423305.

The authors are with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102. Roberto Rojas-Cessa is the corresponding author. Email: rojasces@njit.edu.

in a 1024×1024 switch, these schemes would consider a scheduler that can match 1024 input ports to 1024 output ports at once. However, the scheduler for such match may be complex to implement [2].

In this paper, we propose a two-stage switch, called two-stage Clos-network switch, as a solution for very-large scale switches. The configuration of this novel two-stage switch is based on our proposed module-first matching scheme that considers a three-stage Clos-network switch as reference. The configuration process of three-stage IQ Clos-network switches is simplified by applying a similar concept of Clos networks, used to reduce the hardware complexity of large scale switches, to the matching process. We propose to perform matching between modules in the first and third stages in the first phase, and matching between input and output ports of those matched modules, afterwards. We call this approach module-first matching (MoM). We use longest queue-occupancy first selection as a weighted scheme to estimate the switching performance when using this simple configuration approach. The combination of a weighted selection scheme with MoM is named WMoM here. We show that MoM reduces the matching size of IQ Clos-network switches, such that small schedulers can be used for very large scale switch, of up to Exabit capacity. As in the example above, a switch with 1024 ports, and n=m=k=32, the largest matching size performed by MoM is 32 instead of 1024, and a 32 × 32 scheduler is feasible to implement. We use module matching to determine the configuration of the second-stage modules and port matching for the configuration of the first-stage modules. With the configuration of the first and second stage modules, the third stage-modules become needless and the architecture becomes a two-stage switch.

The remainder of this paper is organized as follows. Section II presents the three-stage Clos-network switch used in the description of module-first matching scheme, and introduces our two-stage Clos-network packet switch. Section III describes the proposed module-first matching scheme. Section IV discusses the implementation of the proposed approach and shows why two stages are sufficient as a Clos-network packet switch. Section V presents the performance study. Section VII presents our conclusions.

II. Three- and Two-Stage Clos-network Switches

The three-stage IQ Clos-network switch is uses virtual output queues (VOQs) in the input ports, as Figure 1.a shows. We use a terminology similar to that in [3], which is as follows:

- $IM(i)$: $(i+1)$th input module, where $0 \leq i \leq k-1$.
- $CM(r)$: $(r+1)$th central module, where $0 \leq r \leq m-1$.
- $OM(j)$: $(j+1)$th output module, where $0 \leq j \leq k-1$.
- n: number of input/output ports in each IM/OM, respectively.
- k: number of IMs/OMs.
- m: number of CMs.
- $IP(i,g)$: $(g+1)$th input port (IP) at $IM(i)$, where $0 \leq g \leq n-1$.
- $OP(j,h)$: $(h+1)$th output port (OP) at $OM(j)$, where $0 \leq h \leq n-1$.
- $VOQ(i,g,j,h)$: Virtual output queue at $IP(i,g)$ that destined for $OP(j,h)$.

There are k input modules (IM), m central modules (CM), and k output modules (OM) in the switch. IMs have a dimension of $n \times m$, OMs have a dimension of $m \times n$, and CMs have a dimension of $k \times k$. The input ports at $IM(i)$ are denoted as $IP(i,g)$. The output ports of $OM(j)$ are denoted as $OP(j,h)$. Each $IP(i,g)$ has $N = n \times k$ VOQs to avoid head-of-line (HOL) blocking. A $VOQ(i,g,j,h)$ stores cells going from $IP(i,g)$ to $OP(j,h)$.

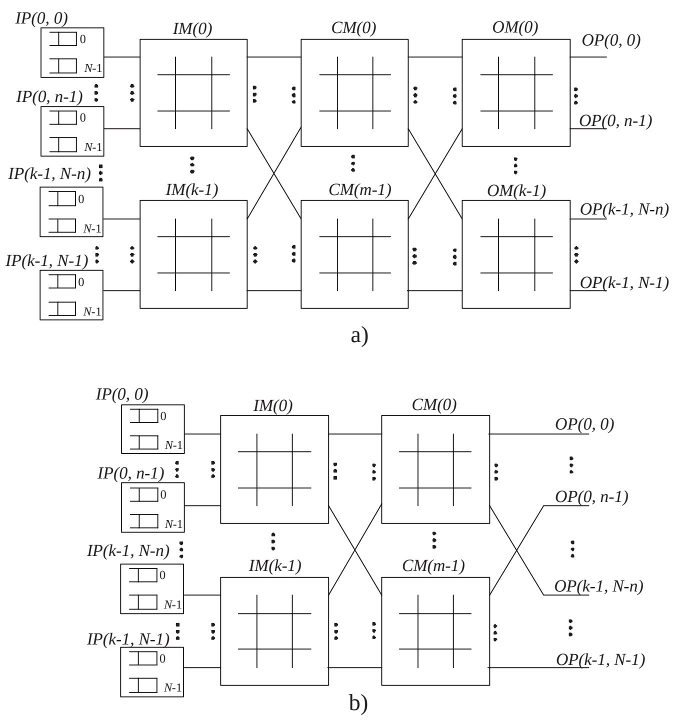

Fig. 1. Clos-network switches: a) the 3-stage switch and b) the 2-stage switch.

Figure 1.b shows our proposed two-stage Clos-network switch, which uses the same notation of the three-stage switch. However, since the third stage is removed, the OMs are not used (for the sake of description we leave the CM labels as in the three-stage switch, although it is clear that we could rename them as OMs). We call this architecture two-stage Clos-network switch as it can be derived from the original three-stage Clos-network.

III. Weight-Based Module-First Matching (WMoM) Scheme

In this section, we introduce a matching scheme for a three-stage Clos-network switch, with however; the purpose of configuring the IMs and CMs of the two-stage Clos-network switch. The WMoM scheduler consists of two classes of schedulers, one class is the module matching scheduler, determining the $IM(i)$-$OM(j)$ pairs to be matched each time slot. The other class is the port matching scheduler, determining the $VOQ(i,g,j,h) - OP(j,h)$ pairs to be matched after the $IM - OM$ pairs are determined. The WMoM is based on a maximal-weight matching selection: longest queue-occupancy first, which follows the iterative longest-queue occupancy first algorithm [12] for single-stage switches. However, other weight-based selection schemes (e.g., oldest cell first) can also be used.

In order to determine $IM(i)$-$OM(j)$ matching, we use a VOQ module counter (VMC) to store the number of cells in $IM(i)$ going to $OM(j)$. A VMC is denoted as

$VMC(i,j)$. In MoM, after module-first matching, we determine the $VOQ(i,g,j,h)$-$OP(j,h)$ matching. Each of the matching processes follow a request-grant-accept approach. The following is the description of MoM:

Part 1: Module matching

- First iteration

- Step 1. $Request$. Each VMC, whose count is larger than zero, sends a request to each output module arbiter[1] for which the IM (i.e., IP) has at least one cell. A request indicates the number of cells for that OM.
- Step 2. $Grant$. If an unmatched OM receives any requests, it chooses the one with the largest occupancy.
- Step 3. $Accept$. If an unmatched IM receives one or more grants, it accepts the one with the largest occupancy.

- ith iteration

- Step 1: Each unmatched VMC sends a request to all unmatched output modules, as in the first iteration.
- Steps 2 and 3: The same procedure is performed as in the first iteration among unmatched VMCs and output modules arbiters.

- Part 2: Port matching.

After Part 1 is complete, we can find the matched modules $IM(i)$ and $OM(j)$ and perform matching among ports for the input and output ports in those modules.

- Step 1 (Request): Each nonempty VOQ of the matched $IM(i)$ sends a request to each output port of the matched $OM(j)$ for which it has a queued cell, indicating the number of cells in that VOQ.
- Steps 2 (grant) and 3 (accept): The same procedure as in the module matching is performed for matching nonempty VOQs of a matched $IM(i)$ and OPs of a matched $OM(j)$, among however, input port arbiters and output port arbiters. These arbiters select the requests with the largest occupancy.

- ith iteration

- Step 1: Each unmatched VOQ in $IM(i)$ at the previous iterations sends another request to all unmatched OP in the matched $OM(j)$ as in Step 1 of the first iteration.
- Steps 2 and 3: The same procedure is performed as in the first iteration for matching between unmatched nonempty VOQs and unmatched output port in the matched $IM(i) - OM(j)$ pairs.

IV. Implementation of a Two-Stage Clos-Network Switch using MoM

The first objective of MoM is to provide a feasible solution for performing the matching processes used to configure an IQ Clos-network switch. For this, the IMs and OMs are first matched, and for that we consider a module scheduler that performs a $k \times k$ matching, and therefore, the module scheduler has k input arbiters and k output arbiters. Since $k = \frac{N}{n}$, the size of the scheduler can be small. The same is the case for the scheduler that performs matching for the input ports of the matched IM to the output ports of the matched OM, called port scheduler. This scheduler performs a $n \times n$ matching, and therefore, it has n input arbiters and n output arbiters. There is one port scheduler in each IM and there is only one module scheduler that can be placed in one of the CMs, where IMs' requests would converge, in a distributed implementation of MoM.

[1]It is not necessary to place an output module arbiter in OMs, they can also be in CMs.

A second property of MoM is the reduced number of information exchange between input ports and the module scheduler, for any number of iterations that the matching process performs. The way the information flows through the switch to perform MoM is as follows: 1) the inputs send a request to the module scheduler, 2) the module scheduler performs module matching, and if several iterations are needed, the module scheduler can perform that without using another requests from the input ports, 3) the module scheduler sends the grants to port schedulers at IMs, and to CMs (and OMs) for their configuration, 4) the port schedulers at IMs perform matching with any number of iterations, and 5) the port schedulers send a grant to the input ports, one per port.

Because an IM is only matched to a single OM, then all CMs have the same configuration at a given time slot. Therefore, the information coming from the module scheduler to all CMs is the same.

Figure 2 shows an example of configurations of a 9×9 three-stage Clos-network switch after the MoM process. All CMs use the same configuration obtained through module matching. In this example, $IM(0)$ is matched to $OM(1)$, $IM(1)$ is matched to $OM(2)$, and $IM(2)$ is matched to $OM(1)$. In $IM(0)$, $IP(0,0)$ is matched to $OP(1,2)$, $IP(0,1)$ is matched to $OP(1,1)$, and $IP(0,2)$ is matched to $OP(1,0)$. Because port matching involves only those IM-CM pairs, the configuration for such match can be done at the IMs only. As shown in this example, OMs would be using always the same configuration (no reconfiguration independently of the matching result), and therefore switching is not performed by them. Therefore, the three-stage switch used in the matching process is only a reference, and a two-stage Clos-network switch (whose modules are indicated by the bold line) suffices.

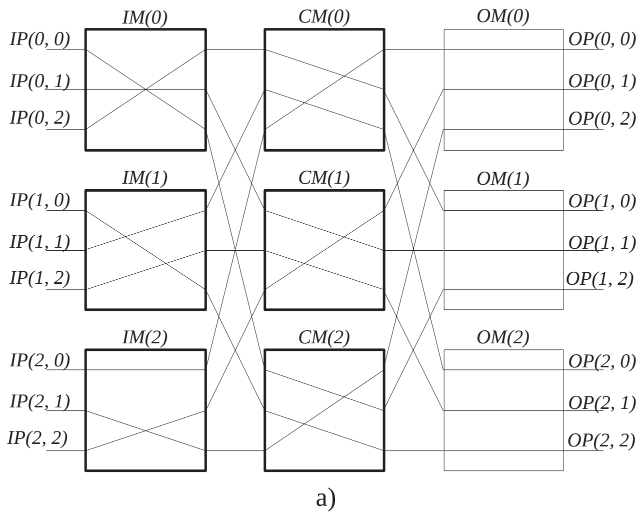

Fig. 2. Example of configuration of the Central Modules.

The module and port arbiters might have counters to retain the number of cells in an input and VOQs. In this way, a single request can be sent from each VOQ to the schedulers.

V. Performance Evaluation

The performance evaluations are produced through computer simulation. We consider WMoM with multiple iterations in module-to-module and port-to-port matching, as the longest queue occupancy matching scheme can give high throughput when a number of iterations of $min\ \{a, b\}$, where a and b is the number of the bipartite members in the matching (e.g., the number of ports or modules). To analyze the effect of the adopted selection scheme, we examine the performance of WMoM with the same number of iterations for module matching and port matching. Specifically, we study a symmetric Clos-network switch with $n = m = k = 8$ and the number of iterations, denoted as Itr, equal to 1, 4, and 8. We selected these values as the adopted matching scheme delivers the lowest performance with $Itr = 1$ and the highest performance whit $Itr = 8$, which is equal to n. The traffic models considered have destinations with uniform and nonuniform distributions and Bernoulli and bursty arrivals. The simulation does not consider the segmentation and re-assembly delays for variable size packets. Simulation results are obtained with a 95% confidence interval, not greater than 5% for the average cell delay.

A. Uniform Traffic

Figure 3 shows the simulation results of WMoM with single and multiple iterations for WMoM under uniform traffic with Bernoulli arrivals. This figure shows that WMoM has a low throughput when both the module matching and port matching processes perform a single iteration. As the number of iterations increases, the throughput of WMoM also increases. This figure shows that WMoM can deliver 100% throughput with $Itr = 8$ under Bernoulli uniform traffic.

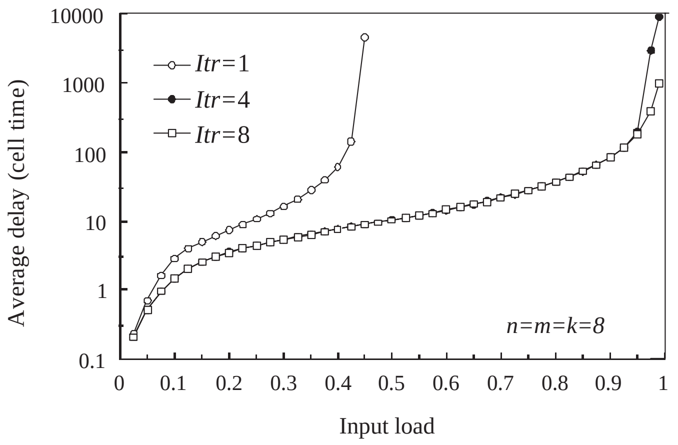

Fig. 3. Average cell delay of WMoM scheme in a *n=m=k=8* switch under Bernoulli uniform traffic.

The bursty traffic follows an on-off Markov modulated process, where the average burst length l is set to 16 cells. Figure 4 shows that WMoM provides low throughput with a single iteration, and above 96% throughput for $Itr = 4$ and $Itr = 8$ under bursty uniform traffic. To improve the throughput under this traffic model, a round-robin selection scheme can be used.

B. Nonuniform Traffic

We simulated the WMoM scheme with multiple iterations under three different nonuniform traffic patterns: unbalanced [13], Chang's [14], asymmetric [15] and diagonal [6].

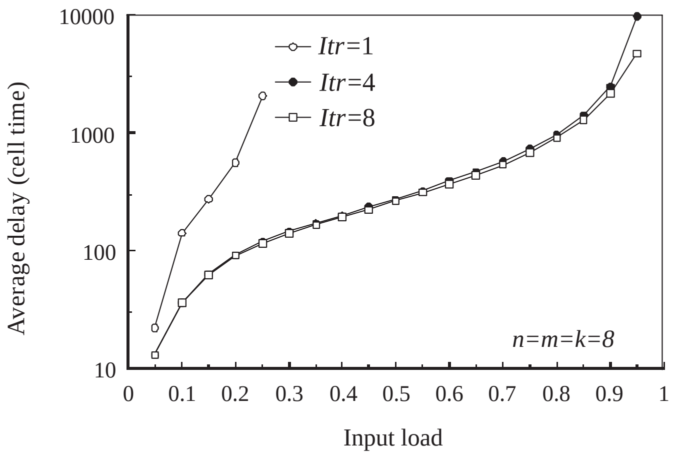

Fig. 4. Average cell delay of WMoM scheme in a *n=m=k=8* under bursty traffic.

The unbalanced traffic model uses a probability, w, as the fraction of input load directed to a single predetermined output, while the rest of the input load is directed to all outputs with uniform distribution. Let us consider IP s, OP d, and the offered input load for each IP ρ. The traffic load from IP s to OP d, $\rho_{s,d}$ is given by,

$$\rho_{s,d} = \begin{cases} \rho\left(w + \frac{1-w}{N}\right) & \text{if } s = d \\ \rho\frac{1-w}{N} & \text{otherwise.} \end{cases} \quad (1)$$

Where N(i.e., nk) is the switch size. When $w = 0$, the offered traffic is uniform. On the other hand, when $w = 1$, the traffic is completely directional. This means that all traffic of IP s is destined for OP d, where $s = d$.

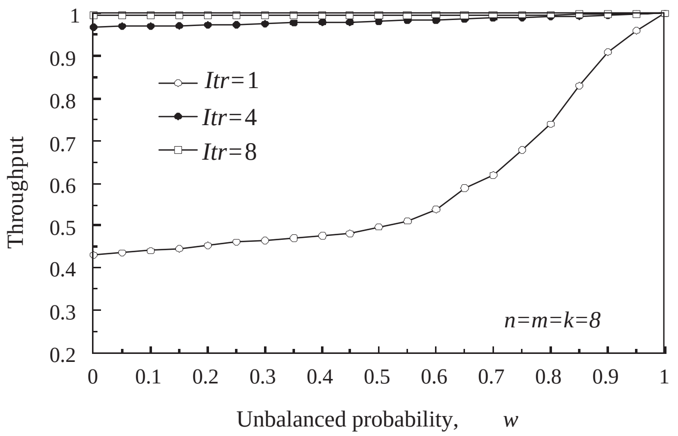

Fig. 5. Throughput performance of WMoM in a *n=m=k=8* switch under unbalanced traffic.

Figure 5 shows the throughput performance of WMoM under unbalanced traffic. This figure shows that WMoM has low throughput with 1 iteration. When the number of iterations increases, the throughput of WMoM increases under the complete range of w. Figure 5 also shows that WMoM has above 97% throughput with 4 iterations and achieves above 99% with 8 iterations. The reason for the improvement shown by WMoM is that n iterations guarantee that n ports are matched, if there is a cell destined to them, as has been observed on single-stage switches.

Another nonuniform traffic pattern is Chang's traffic model, which is defined as $\rho_{i,j} = 0$ when $i = j$, and $\rho_{i,j} = 1/(N-1)$, otherwise, where $N = nk$ and $\rho_{i,j}$ is the input load. Although the performance graph is not shown, the WMoM scheme delivers similar performance, in terms of throughput and delay, as that for uniform traffic.

The asymmetric traffic model is described in [15]. This traffic model has a different distribution of the input load for each output port. Figure 6 shows that WMoM delivers low throughput, even with multiple iterations, under asymmetric traffic, as the throughput barely reaches 75% with $Itr = 8$.

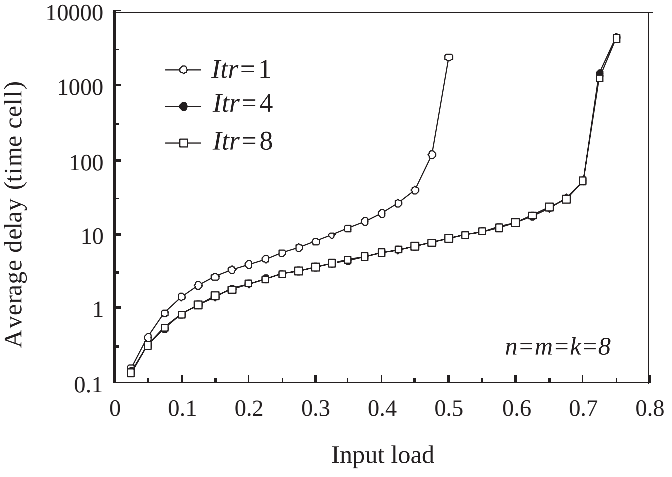

Fig. 6. Average cell delay of WMoM scheme in a *n=m=k=8* switch under asymmetric traffic.

We also tested WMoM under the diagonal traffic model, which is defined here as $\rho_{i,j} = 2/3$ when $j = i$, $\rho_{i,j} = 1/3$ when $j = (i + 1)$ modulo N, and $\rho_{i,j} = 0$ otherwise. Figure 7 shows that WMoM has lower throughput than that obtained under asymmetric traffic as this traffic model has a strong nonuniform distribution among only two output ports per input. One approach that can be used to improve the throughput of this switch is by applying framed matching schemes [19].

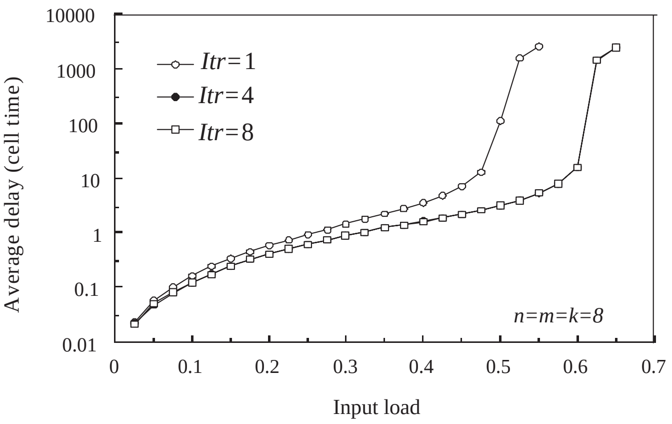

Fig. 7. Average cell delay of WMoM in a *n=m=k=8* switch under diagonal traffic.

VI. Scalability

The reduction of scheduler sizes by module matching allows to consider the implementation of large switches. We consider two different strategies: a) with n=k=m, and b) with a more flexible selection of n and m values. Table I shows an example of the component size for switches with n=k=m. Here, the size of the IMs/OMs and CMs are denoted as $|IM|$ and $|CM|$, respectively. The number of module schedulers, denoted as MS, is always one, and the number of port schedulers, denoted as PS, is k. The sizes of the module and port schedulers are denoted as $|MS|$ and $|PS|$, respectively.

N	n=m=k	$\lvert IM\rvert$ = $\lvert CM\rvert$	PSs	$\lvert MS\rvert$	$\lvert PS\rvert$
256	16	16x16	16	16	16
1024	32	32x32	32	32	32
4096	64	64x64	64	64	64
16384	128	128x128	128	128	128

TABLE I
EXAMPLE OF SCHEDULER SIZES OF CLOS-NETWORK SWITCHES WITH $n = k = m$.

N	n	k	m	MSs	PSs	$\lvert MS\rvert$	$\lvert PS\rvert$
256	64	4	4	1	4	4	64
512	64	8	8	1	8	8	64
1024	64	16	16	1	16	16	64
2048	64	32	32	1	32	32	64
4096	64	64	64	1	64	64	64
8192	64	128	128	1	128	128	64

TABLE II
EXAMPLE OF SCHEDULER SIZES IN A SWITCH WITH FLEXIBLE CONFIGURATION.

Here, we consider that the maximum matching size is 64 to reduce hardware and time complexities. Since the implementation issues related to cabling and distribution of a large number of chips is out of the scope of this paper, we consider that large quantities of such elements are acceptable.

For switches with n=k=m, the number of size possibilities is rather reduced, so we can consider a more flexible selection of n and m as Table II shows.

By the two tables above, it is clear that the switch size is limited to 4096 ports with a matching size of 64 (i.e., 64$\times$64 schedulers). A larger number of ports would increase the size of module schedulers and the CMs, beyond our restricted value in our example. However, the module matching principle can be applied to nested Clos-network switches, used to reduce the CM sizes.

A. Nested Switches and Recursive Module-First Matching

Nested Clos-network switches can be seen as a recursive application of the Clos-network configuration directly into any module (e.g., IM, CM, or OM in a three-stage switch, and IM and CM in a two-stage switch) of a switch. For the sake of simplicity, let's consider that nesting is applied to CMs, and that only two levels are used (i.e., a CM has one Clos-network configuration within and the modules inside are only single-stage switches), as Figure 8 shows. This figure also shows the order the matching process follows in nested Clos-network switches, first the module matching of the internal modules in CM (modules with bold lines), then the IM-OM modules external to the CMs (modules with bold-dashed lines), and finally the port matching among matched IM-OM pairs (ports with bold-dashed lines) at $IM(k-n-1)$ and $OM(n-1)$. The nested two-stage Clos-network switch is indicated by the

N	8192	16384	32768	65536	262144
n	64	64	64	64	64
k	128	256	256	1024	4196
m	64	64	64	64	64
g_{MS}	2	4	4	16	64
$\|g_{MS}\|$	2	4	4	16	64
MSs	2	4	4	16	64
PSs	128	256	256	1024	4196
$\|MS\|$	64	64	64	64	64
$\|PS\|$	64	64	64	64	64

TABLE III
SCHEDULER SIZES OF NESTED CLOS-NETWORK SWITCHES.

large rectangle in this figure. Therefore, the architecture of CMs can use a Clos-network configuration.

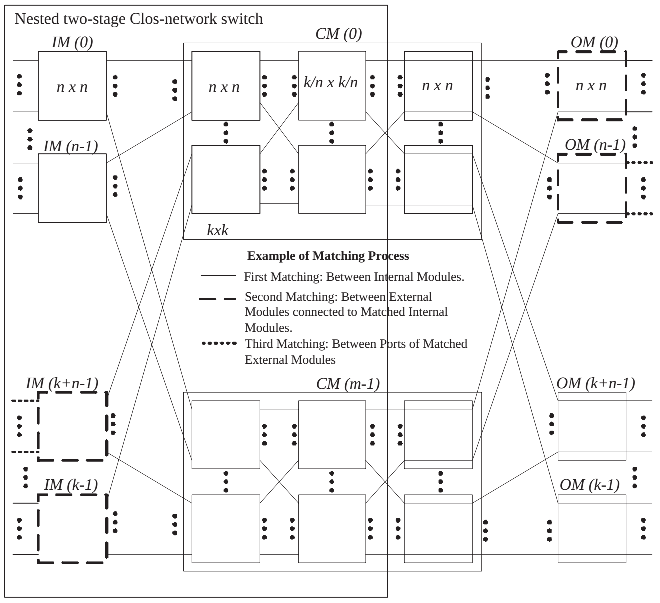
Nested two-stage Clos-network switch
IM (0)
n x n
IM (n-1)
CM (0)
n x n
k/n x k/n
n x n
kxk
OM (0)
n x n
OM (n-1)
Example of Matching Process
First Matching: Between Internal Modules.
Second Matching: Between External Modules connected to Matched Internal Modules.
Third Matching: Between Ports of Matched External Modules
IM (k+n-1)
CM (m-1)
OM (k+n-1)
IM (k-1)
OM (k-1)

Fig. 8. Nested Clos-network switch for large scale packet switches.

Table VI-A shows the hardware count of several examples of nested Clos-networks using module-first matching. The scheduler that performs module matching in the internal Clos is denoted as g_{MS}, and the size of g_{MS} as $|g_{MS}|$. Here, it can be seen with a restricted scheduler size of 64, the maximum port count is up to 262,144. In a packet switch with 160-Gbps ports, module-first matching would allow to configure a 40 Ebps (Exabit per second) switch, resolved in three phases as indicated above.

VII. CONCLUSIONS

In this paper, we proposed a two-stage Clos-network switch for scalable IQ Clos-network packet switches. The packet switching fashion allows us to consider the reduction of the original Clos-network switch proposed for circuit switching. This novel two-stage switch uses a configuration scheme that considers a three-stage Clos-network switch. The proposed scheme uses module-first matching. This scheme matches a single IM to a single OM to reduce the configuration complexity of the switch. Therefore, module matching is performed before port matching, and this is the main the difference from the existing schemes. As an example, we used a weighted selection scheme based on the longest VOQ occupancy to show the performance of this switch under module-matching first. The scheduler complexity for implementing MoM is reduced by using a similar concept to that of Clos networks, used to reduce the hardware amount of a large switch, to the matching process. We showed that for a large scale switch, of up to Exabit capacity, would use a small scheduler size.

This paper also showed that WMoM, using longest occupancy-queue first provides 100% throughput under Bernoulli uniform traffic for a 64×64 switch, and above 99.5% throughput under Bernoulli unbalanced traffic, respectively. Furthermore, the proposed scheme shows high and also low performance under other nonuniform traffic patterns. These results are, to the best of our knowledge, the only ones presented for input-queue Clos-network switches under diagonal, asymmetric, Chang's traffic models, and these can be used for comparison in future research.

REFERENCES

[1] C. Clos," A study of nonblocking switching networks," *Bell Syst. Tech. J.*, pp. 406-424, Mar 1953.
[2] P. Gupta and N. McKeown, "Design and Implmentation of a Fast Crossbar Scheduler," in Proc. of *Hot Interconnects VI*, Stanford University, August 1998.
[3] E. Oki, Z. Jing, R. Rojas-Cessa, and H. J. Chao, "Concurrent roun-robin dispatching scheme for Clos-network switches," *IEEE/ACM Trans. Networking*, vol. 10, no. 6, pp. 830-844, May 2001.
[4] F.M. Chiussi, J.G. Kneuer, and V.P. Kumar "Low-cost scalable switching solutions for broadband networking: the ATLANTA architecture and chipset " *IEEE Commun. Mag.*, pp. 44-53, Dec. 1997.
[5] R. Rojas-Cessa, E. Oki, and H. J. Chao, "Maximum weight matching dispatching scheme in buffered Clos-network packet switches," in Proc. *IEEE International Conference on Communications* , Vol. 2, pp. 1075-1079, June. 2004.
[6] K. Pun and M. Hamdi, "Static round-robin dispatching schemes for Clos-network switches," *IEEE Workshop on High Performance Switching and Routing*, pp. 239-243, May 2002.
[7] M. Karol, M. Hluchyj, "Queuing in High-performance Packet-switching," *IEEE J. Select. Area Commun.*, vol. 6, pp. 1587-1597, December 1988.
[8] T. T. Lee, and S-Y Liew, "Parallel Routing Algorithm in Benes-Clos Networks," in Proc. *IEEE INFOCOM' 96*, pp. 279-286, 1996.
[9] H.J. Chao, S.Y. Liew, and Z. Jing, "A dual-level Matching algorithm for 3-stage Clos-network Packet Switches," *IEEE 11th Symposium on High Performance Interconnects*, pp. 38-44, August 2003.
[10] H.J. Chao, Z. Jing; S.Y. Liew, "Matching algorithms for three-stage bufferless Clos network switches," *IEEE Commun. Mag.*, Vol. 41, Issue 10, pp. 46-54, Oct. 2003.
[11] K. Pun and M. Hamdi, "Distro: A Distributed Static Round-robin Schedulinf Algorithm for Bufferless Clos-network switches," in Proc. *IEEE Globecom 2002*, Vol. 3, pp. 2298-2302, 2002.
[12] N. McKeown, A. Mekkittikul, V. Anantharam, J. Walrand, "Achieving 100% Throughput in an Input-queued Switch," *IEEE Trans. Commun.*, Vol. 47, No. 8, pp. 1260-1267, August 1999.
[13] R. Rojas-Cessa, E. Oki, Z. Jing, and H. J. Chao, "CIXB-1: Combined Input-One-cell-crosspoint Buffered Switch," in Proc. *IEEE HPSR 2001*, pp. 324-329, May 2001.
[14] C-S. Chang, D-S. Lee, and Y-S. Jou, "Load Balanced Birkhoff-von Newman Switches," in Proc. *IEEE HPSR 2001*, pp.276-280, April 2001.
[15] R. Schoene, G. Post and G. Sander, "Weighted Arbitration Algorithms with Priorities for Input-Queued Switches with 100% Throughput," *Broadband Switches Symposium'99*, 1999. http://www.schoenen-service.de/assets/papers/Schoenen99bssw.pdf
[16] N. McKeown, "The iSLIP scheduling algorithm for Input-queued Switches," *IEEE/ACM Trans. Networking*, Vol. 7, No. 4, pp. 188-201, April 1999.
[17] C-S. Chang, W-J. Chen and H-Y. Huang, "Birkhoff-von Neumann input buffered crossbar switches," in Proc. *IEEE INFOCOM 2000*, Vol. 3, pp. 1614-1623, March 2000.
[18] J. Turner, and N. Yamanaka, "Architectural Choices in Large Scale ATM switches," *IEICE Trans. Commun.*, Vol. E81-B, no. 2, pp. 120-137, Feb. 1998
[19] C-B. Lin and R. Rojas-Cessa, "Frame Occupancy-Based Dispatching Schemes for Buffered Clos-Network Packet Switch," in Proc. of *IEEE International Conference on Networks*, Kuala Lumpur, Malaysia, Nov. 2005.