7

Packet Dispatching Schemes for Three-Stage Buffered Clos-Network Switches

Janusz Kleban
Poznan University of Technology
Poland

1. Introduction

The continued growth of Internet Protocol-based traffic like data, voice, audio, TV, and gaming traffic, requires much more robust, highly scalable core routers/switches to handle the expected annual doubling of bandwidth in the United States and Europe and the expected tripling and possibly quadrupling of bandwidth in Asia. In the near future service providers will need to deploy a new class of core routers that have taken a major leap forward in design. While the bandwidth of external connections on core routers has increased in recent years from STM-1 to STM-16 and STM-64, tomorrow's core routers will need to support STM-256 connections operating at 40 Gbps. In addition, the number of line cards that the core router will need to support will grow dramatically to handle the aggregate subscriber and backbone bandwidth. To meet these new demands, tomorrow's router architectures will have to function very differently from those of today. They will require distributed memories and multi-stage switching fabrics that replace single-stage crossbars, allowing extraordinary scalability.
The main part of each high-performance network node is a switching fabric - instead of a shared central bus - which transfers a packet from its input link to its output link (Fig. 1). The switching fabric provides very fast transmission between line cards, therefore the router throughput is improved. Internally, high capacity switches/routers operate on fixed-size data units, called cells from the ATM jargon. This means that in the case of variable-size packets on transmission lines, as it is normally the case in the Internet, packets must be segmented into cells at switch inputs, and cells must be reassembled into packets at switch outputs (Chao & Cheuk, 2001). There are mainly two approaches to the implementation of high-speed packet switching systems. One approach is the single-stage switch architecture such as the crossbar switch, the other one is the multiple-stage switch architecture, such as the Clos-network switch. Most high-speed packet switching systems in the backbone of the Internet are currently built on the basis of a single-stage switching fabric with a centralized scheduler. Crossbar switches are internally nonblocking and simple in architecture. However, they are only little scalable due to the number of the crosspoints, which grows as N^2, where N is the total number of inputs/outputs. Multiple-stage Clos-network switches are a potential solution to overcome the limited scalability of single-stage switches, in terms of the number of input/output chip pins and the number of switching elements.

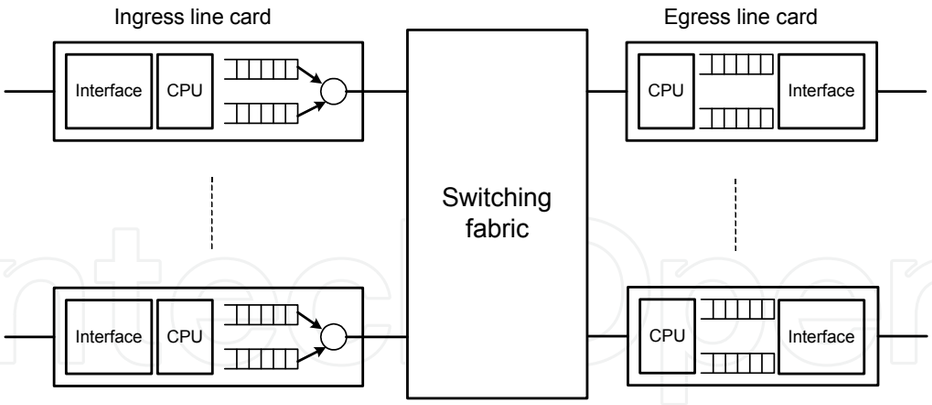

Fig. 1. High-performance router architecture

In the Clos-network switch packet scheduling is needed as there is a large number of shared resources where contention may occur. A cell transmitted within the multiple-stage Clos switching fabric can face internal blocking or output port contention. Internal blocking occurs when two or more cells contend for an internal link at the same time (Fig.2). A switch suffering from internal blocking is called blocking contrary to a switch that does not suffer from internal blocking called nonblocking. The output port contention occurs if there are multiple cells contend for the same output port.

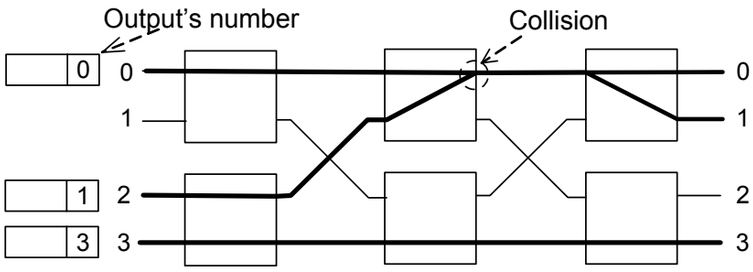

Fig. 2. Internal blocking: two cells destined for output ports 0 and 1 try to go through the same internal link, at the same time

Cells that have lost contention must be either discarded or buffered. Generally speaking, buffers may be placed at inputs, outputs, inputs and outputs, and/or within the switching fabric. Depending on the buffer placement respective switches are called input queued (IQ), output queued (OQ), combined input and output queued (CIOQ) and combined input and crosspoint queued (CICQ) (Yoshigoe &Christensen, 2003).
In the OQ strategy all incoming cells (i.e. fixed-length packets) are allowed to arrive at the output port and are stored in queues located at each output of switching elements. The cells destined for the same output port simultaneously do not face a contention problem because they are queued in the buffer at the output. To avoid the cell loss the system must be able to write N cells in the queue during one cell time. No arbiter is required because all the cells can be switched to respective output queue. The cells in the output queue are served using FIFO discipline to maintain the integrity of the cell sequence. In OQ switches the best performance (100% throughput, low mean time delay) is achieved, but every output port must be able to accept a cell from every input port simultaneously or at least within a single

time slot (a time slot is the duration of a cell). An output buffered switch can be more complex than an input buffered switch because the switching fabric and output buffers must effectively operate at a much higher speed than that of each port to reduce the probability of cell loss. The bandwidth required inside the switching fabric is proportional to both the number of ports *N* and the line rate. The internal speedup factor is inherent to pure output buffering, and is the main reason of difficulties in implementing switches with output buffering. Since the output buffer needs to store *N* cells in each time slot, its speed limits the switch size.

The IQ packet switches have the internal operation speed equal to (or slightly higher) than the input/output line speed, but the throughput is limited to 58,6% under uniform traffic and Bernoulli packet arrivals because of Head-Of-Line (HOL) blocking phenomenon (Chao & Cheuk, 2001). HOL blocking causes the idle output to remain idle even if at an idle input there is a cell waiting to be sent to an (idle) output. Due to other cell that is ahead of it in the buffer the cell cannot be transmitted over the switching fabric. An example of HOL blocking is shown in Fig. 3. This problem can be solved by selecting queued cells other than the HOL cell for transmission, but it is difficult to implement such queuing discipline in hardware. Another solution is to use speedup, i.e. the switch's internal links speed is greater than inputs/outputs speed. However, this also requires a buffer memory speed faster than a link speed. To increase the throughput of IQ switches space parallelism is also used in the switch fabric, i.e. more than one input port of the switch can transmit simultaneously.

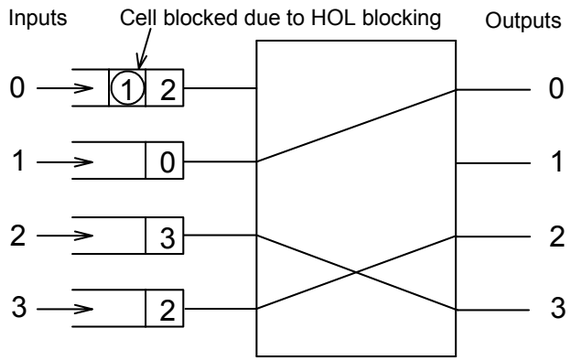

Fig. 3. Head-of-line blocking

The virtual output queuing (VOQ) is widely implemented as a good solution for input queued (IQ) switches, to avoid the HOL blocking encountered in the pure input-buffered switches. In VOQ switches every input provides a single and separate FIFO for each output. Such a FIFO is called a Virtual Output Queue. When a new cell arrives at the input port, it is stored in the destined queue and waits for transmission through a switching fabric.

To solve internal blocking and output port contention issues in VOQ switches fast arbitration schemes are needed. An arbitration scheme is essentially a service discipline that arranges the transmission order among the input cells. It decides which items of information should be passed from inputs to arbiters, and – based on that decision – how each arbiter picks one cell from among all input cells destined for the output. The arbitration decisions for every output port have to be taken in each time slot using a central arbiter, or distributed arbiters. In the distributed manner, each output has its own arbiter operating independently from others. However, in this case it is necessary to send many request-grant-accept signals.

It is very difficult to implement such arbitration in the real environment because of time constraints. A central arbiter may also create a bottleneck due to time constraints as the switch size increases.
Considerable work has been done on scheduling algorithms for the crossbar and three-stage Clos-network VOQ switches. Most of them achieve 100% throughput under the uniform traffic, but the throughput is usually reduced under the nonuniform traffic (Chao & Liu, 2007). A switch can achieve 100% throughput under the uniform or nonuniform traffic if the switch is stable, as it was defined in (McKeown at al., 1999). In general, a switch is stable for a particular arrival process if the expected length of the input queues does not grow without limits.
This chapter presents basic ideas concerning packet switching in next generation switches/routers. The simulation results obtained by us for the well known and new packet dispatching schemes for the three-stage buffered Clos-network switches are also shown and discussed. The remainder of the chapter is organized as follows: subchapter 2 introduces some background knowledge concerning the Clos-network switch that we refer to throughout this chapter; subchapter 3 presents packet dispatching schemes with distributed arbitration; subchapter 4 is devoted to dispatching schemes with centralized arbitration. A survey of related works is carried out in subchapter 5.

2. Clos switching network

In 1953, Clos proposed a class of space-division three-stage switching networks and proved strictly non-blocking conditions of such networks (Clos, 1953). These kind of switching fabrics are widely used and extensively studied as a scalable and modular architecture for the next generation switches/routers. The Clos switching fabric can achieve a nonblocking property with the smaller number of total crosspoints in the switching elements than crossbar switches. Nonblocking switching fabrics are divided into four classes: strictly nonblocking (SSNB), wide-sense nonblocking (WSNB), rearrageable nonblocking (RRNB) and repackably nonblocking (RPNB) (Kabacinski, 2005). SSNB and WSNB ensures, that any pair of idle input and output can be connected without changing any existing connections, but a special path set-up strategy must be used in WSNB networks. In RRNB and RPNB any such pair can be also connected, but it may be necessary to re-switch existing connections to other connecting paths. The difference is in time these reswitchings take place. In RRNB, when a new request arrives, and is blocked, an appropriate control algorithm is used to reswitch some of existing connections to unblock the new call. In RPNB, a new call can always be set up without reswitching of existing connections, but reswitching takes place when any of existing call is terminated. These reswitchings are done to prevent a switching fabric from blocking states before a new connection arrives.
The three-stage Clos-network architecture is denoted by $C(m, n, k)$, where parameters m, n, and k entirely determine the structure of the network. There are k input switches of capacity $n \times m$ in the first stage, m switches of capacity $k \times k$ in the second stage, and k output switches of capacity $m \times n$ in the third stage. The capacity of this switching system is $N \times N$, where $N = nk$. The three-stage Clos switching fabric is strictly nonblocking if $m \geq 2n-1$ and rearrangeable nonblocking if $m \geq n$. The three-stage Clos-network switch architecture may be categorized into two types: bufferless and buffered. The former one has no memory in any stage, and it is also referred to as the Space-Space-Space (S^3) Clos-network switch, while

the latter one employs shared memory modules in the first and third stages, and is referred to as the Memory-Space-Memory (MSM) Clos-network switch. The buffers in the second stage modules cause an out-of-sequence problem, so a re-sequencing function unit in the third stage modules is necessary but difficult to implement when the port speed increases. One disadvantage of the MSM architecture is that the first and third stages are both composed of shared-memory modules.

We define the MSM Clos switching fabric based on the terminology used in (Oki at al., 2002a) (see Fig. 4 and Table 1).

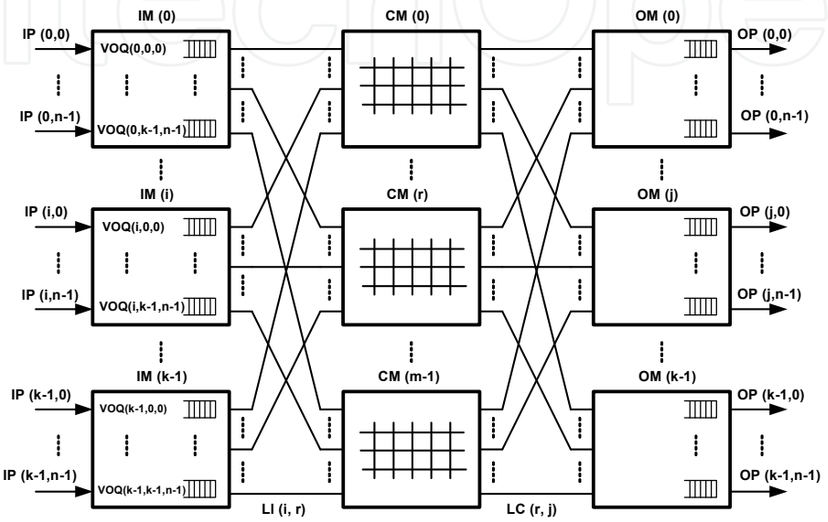

Fig. 4. The MSM Clos switching network

Notation	Description
IM	Input module at the first stage
CM	Central module at the second stage
OM	Output module at the third stage
i	IM number, where $0 \le i \le k-1$
j	OM number, where $0 \le j \le k-1$
h	Input/output port number in each IM/OM, where $0 \le h \le n-1$
r	CM number, where $0 \le r \le m-1$
IM (i)	The *(i+1)*th input module
CM (r)	The *(r+1)*th central module
OM (j)	The *(j+1)*th output module
IP (i, h)	The *(h+1)*th input port at *IM(i)*
OP (j, h)	The *(h+1)*th output port at *OM(j)*
LI (i, r)	Output link at *IM(i)* that is connected to *CM(r)*
LC (r, j)	Output link at *CM(r)* that is connected to *OM(j)*
VOQ (i, j, h)	Virtual output queue that stores cells from *IM(i)* to *OP(j, h)*

Table 1. A notation for the MSM Clos switching fabric

In the MSM Clos switching fabric architecture the first stage consists of k IMs, and each of them has an $n \times m$ dimension and nk VOQs to eliminate Head-Of-Line blocking. The second stage consists of m bufferless CMs, and each of them has a $k \times k$ dimension. The third stage

consists of k OMs of capacity $m \times n$, where each *OP(j, h)* has an output buffer. Each output buffer can receive at most m cells from m CMs, so a memory speedup is required here.
Generally speaking, in the MSM Clos switching fabric architecture each *VOQ(i, j, h)* located in *IM(i)* stores cells going from *IM(i)* to *OP(j, h)* at *OM(j).* In one cell time slot VOQ can receive at most n cells from n input ports and send one cell to any CMs. A memory speedup of n is required here, because the rate of memory work has to be n times higher than the line rate. Each *IM(i)* has m output links connected to each *CM(r)*, respectively. A *CM(r)* has k output links *LC(r, j),* which are connected to each *OM(j)*, respectively.
Input buffers located in IMs may be also arranged as follows:

- An input buffer in each input port is divided into N parallel queues, each of them storing cells directed to different output ports. Each IM has nN VOQs, no memory speedup is required.
- An input buffer in each IM is divided into k parallel queues, each of them storing cells destined to different OMs. Those queues will be called Virtual Output Module Queues (VOMQs), instead of VOQs. It is possible to arrange buffers in such way because OMs are nonblocking. Memory speedup of n is necessary here. In that case, there are less queues in each IMs but they are longer than VOQs. Each *VOMQ(i, j)* stores cells going from *IM(i)* to the *OM(j).*
- Each input of an IM has k parallel queues, each of them storing cells destined to different OMs; we call it mVOMQs (multiple VOMQs). In each IM there are nk mVOMQs. This type of buffer arrangement eliminates a memory speedup. Each *mVOMQ(i, j, h)* stores cells going from *IP(i, h)* to the *OM(j)*, h denotes the input port number or the number of a VOMQ group.

Thanks to allocating buffers in the first and third stages the main switching problem in the three-stage buffered Clos-network switches lies in routes assignment between input and output modules.

3. Packet dispatching algorithms with distributed arbitration

The packet dispatching algorithms are responsible for choosing cells to be sent from the VOQs to the output buffers, and simultaneously for selecting connecting paths from IMs to OMs. Considerable work has been done on packet dispatching algorithms for the three-stage buffered Clos-network switches. Unfortunately, the known optimal algorithms are too complex to implement at very high data rates, so sub-optimal, heuristic algorithms of lesser complexity, but also lesser performance, have to be used. The idea of three-phase algorithm, namely request-grant-accept, described by Hui and Arthurs (Hui & Arthus, 1987), is widely used by the packet dispatching algorithms with distributed arbitration. In this algorithm many request, grant and accept signals are sent between each input and output to do matching. In general, the three-phase algorithm works as follows: each unmatched input sends a request to every output for which it has a queued cell. If an unmatched output receives multiple requests, it grants one over all requests. If an input receives multiple grants, it accepts one and sends an accept signal to matched output. These three steps may be repeated in many iterations.
The primary multiple-phase dispatching algorithms for the three-stage buffered Clos-network switches were proposed in (Oki at al. 2002a). The basic idea of these algorithms is to use the effect of desynchronization of arbitration pointers and common request-grant-

accept handshaking scheme. The well known algorithm with multiple-phase iterations is the CRRD (Concurrent Round-Robin Dispatching). Other algorithms like the CMSD (Concurrent Master-Slave Round-Robin Dispatching) (Oki at al. 2002a), SRRD (Static Round-Robin Dispatching) (Pun & Hamdi, 2004), and proposed by us in (Kleban & Wieczorek, 2006) - CRRD-OG (Concurrent Round-Robin Dispatching with Open Grants) use the main idea of the CRRD scheme and try to improve results by implementing different mechanisms. We start to describe these algorithms with presentation of very simple scheme called Random Dispatching (RD).

3.1 Random dispatching scheme

Random selection as dispatching scheme is used by the ATLANTA switch developed by Lucent Technologies (Chao & Liu, 2007). An explanation of the basic concept of Random Dispatching (RD) scheme should help us to understand how the CRRD and CRRD-OG algorithms work.

The basic idea of RD scheme is quite similar to the PIM (Parallel Iterative Matching) scheduling algorithm used in the single stage switches. In this scheme two phases are considered for dispatching from the first to second stages. In the first phase each IM randomly selects up to m VOQs and assigns them to IM output links. In the second phase requests associated with output links are sent from an IM to a CM. The arbitration results are sent from CMs to IMs, so the matching between IMs and CMs can be completed. If there is more than one request for the same output link in the CM, it grants one request randomly. In the next time slot the granted VOQs will transfer their cells to the corresponding OPs.

In detail, the RD algorithm works as follows:

- *PHASE 1*: Matching within IM:
  - *Step 1:* Each nonempty VOQ sends a request for candidate selection.
  - *Step 2*: The *IM(i)* selects up to m requests out of nk nonempty VOQs. A round-robin arbitration can be employed for this selection.
- *PHASE 2:* Matching between IM and CM:
  - *Step 1:* A request that is associated with *LI(i, r)* is sent out to the corresponding *CM(r)*. An arbiter that is associated with *LC(r, j)* selects one request among k and the *CM(r)* sends up to k grants, each of which is associated with one *LC(r, j)*, to the corresponding IMs.
  - *Step 2:* If the VOQ at the IM receives the grant from the CM, it sends the corresponding cell at the next time slot. Otherwise, the VOQ will be a candidate again at step 2 in Phase 1 at the next time slot.

It has been shown that a high switch throughput cannot be achieved due to the contention at the CM, unless the internal bandwidth is expanded. To achieve 100% throughput the expansion ratio m/n has to be set to at least: $(1–1/e)^{-1} \approx 1{,}582$ (Oki at al. 2002a).

3.2 Concurrent Round-Robin Dispatching

The Concurrent Round Robin Dispatching (CRRD) algorithm has been proposed to overcome the throughput limitation of the RD scheme. The basic idea of this algorithm is to use the desynchronization of arbitration pointers effect in the three-stage Clos-network switch. It is based on common request-grant-accept handshaking scheme and achieves 100%

throughput under uniform traffic. To easily obtain pointers desynchronization effect the *VOQ(i, j, h)* in the *IM(i)* are rearranged for dispatching as follows:

VOQ(i, 0, 0), VOQ(i, 1, 0), VOQ(i, 2, 0), ... , VOQ(i, k-1, 0)
VOQ(i, 0, 1), VOQ(i, 1, 1), VOQ(i, 2, 1), ... , VOQ(i, k-1, 1)
...
VOQ(i, 0, n-1), VOQ(i, 1, n-1), VOQ(i, 2, n-1) ,..., VOQ(i, k-1, n-1)

Therefore, *VOQ(i, j, h)* is redefined as *VOQ(i, v)*, where $v = hk + j$ and $0 \le v \le nk - 1$.

Each *IM(i)* has *m* output link round-robin arbiters and *nk* VOQ round-robin arbiters. An output link arbiter associated with *LI(i, r)* has its own pointer *PL(i, r)*. A VOQ arbiter associated with the *VOQ(i, v)* has its own pointer *PV(i, v)*. In *CM(r)*, there are *k* round robin arbiters, each of which corresponds to *LC(r, j)* – an output link to the *OM(j)* – and has its own pointer *PC(r, j)*.

The CRRD algorithm completes the matching process in two phases. In Phase 1 at most *m* VOQs are selected as candidates, and the selected VOQ is assigned to an IM output link. An iterative matching with round-robin arbiters is adopted within the *IM(i)* to determine the matching between a request from the *VOQ(i, v)* and the output link *LI(i, r)*. This matching is similar to the iSLIP approach (Chao & Liu, 2007). In Phase 2, each selected VOQ that is associated with each IM output link sends a request from an IM to a CM. The CMs respond with the arbitration results to IMs so that the matching between IMs and CMs can be done.

The pointers *PL(i, r)* and *PV(i, v)* in the *IM(i)* and *PC(r, j)* in the *CM(r)* are updated to one position after the granted position, only if the matching within the IM is achieved at the first iteration on Phase 1 and the request is also granted by the CM in Phase 2.

It was shown that there is a noticeable improvement in the cell average delay by increasing the number of iterations in each IM. However, the number of iterations is limited by the arbitration time in advance. Simulation results obtained by us shown that the optimal number of iterations in the IM is *n/2* and more iterations do not produce a measurable improvement.

The CRRD algorithm works as follows:

- *PHASE 1:* Matching within IM

  First iteration:
  - *Step 1:* Request: Each nonempty *VOQ(i, v)* sends a request to every arbiter of the output link *LI(i, r)* within *IM(i)*.
  - *Step 2:* Grant: Each arbiter of the output link *LI(i, r)* chooses one VOQ request in a round-robin fashion and sends the grant to the selected VOQ. It starts searching from the position of *PL(i, r)*.
  - *Step 3:* Accept: Each arbiter of *VOQ(i, v)* chooses one grant in a round-robin fashion and sends the accept to the matched output link *LI(i, r)*. It starts searching from the position of *PV(i, v)*.

  i-th iteration (*i*>1):
  - *Step 1:* Each unmatched *VOQ(i, v)* at the previous iterations sends another request to all unmatched output link arbiters.
  - *Step 2 and 3:* These steps are the same as in the first iteration.
- *PHASE 2:* Matching between IM and CM
  - *Step 1:* Request: Each selected in Phase 1 IM output link *LI(i, r)* sends the request to *CM(r) j*th output link *LC(r, j)*.

- *Step 2:* Grant: Each round-robin arbiter associated with output link *LC(r, j)* chooses one request by searching from the position of *PC(r, j)*, and sends the grant to the matched output link *LI(i, r)* of *IM(i)*.
- *Step 3:* Accept: If the *LI(i, r)* receives the grant from the *LC(r, j)* it sends the cell from the matched *VOQ(i, v)* to the *OP(j, h)* through the *CM(r)* at the next time slot. The IM cannot send the cell without receiving the grant. Not granted requests from the CM will be again attempted to be matched at the next time slot because the round-robin pointers are updated to one position after the granted position only if the matching within IM is achieved in Phase 1 and the request is also granted by the CM in Phase 2.

3.3 Concurrent Round-Robin Dispatching with Open Grants

The Concurrent Round-Robin Dispatching with Open Grants (CRRD-OG) algorithm is an improved version of the CRRD scheme in terms of the number of iterations which are necessary to achieve better results. In the CRRD-OG algorithm a mechanism of open grants is implemented. An open grant is sent by a CM to an IM and contains information about unmatched link from the second to the third stage. In other words, the *IM(i)* is informed about unmatched output link *LC(r, j)* to the *OM(j)*. The open grant is sent by each unmatched output link *LC(r, j)*. Due to the architecture of the three-stage Clos switching fabric is clearly defined, it is also information about output port numbers, which can be reached using the output *j* of the *CM(r)*. On the basis of this information the *IM(i)* looks up through VOQs and searches a cell which is destined to any output of the *OM(j)*. If such cell exists it will be sent at the next time slot. To support the process of searching the proper cell to be sent to the *OM(j)* each IM has *k* open grant arbiters with *POG(i, j)* pointers. Each arbiter is associated with the *OM(j)* accessible by the output link *LC(r, j)* of the *CM(r*). The *POG(i, j)* pointer is used to search VOQs located at each input port according to the round robin routine.

In the CRRD-OG algorithm two phases are necessary to complete matching process. Phase 1 is the same as in the CRRD algorithm. In Phase 2 the CRRD-OG algorithm works as follows:

- *PHASE 2:* Matching between IM and CM
  - *Step 1:* Request: Each selected in Phase 1 IM output link *LI(i, r)* sends the request to the *CM(r) j*th output link *LC(r, j)*.
  - *Step 2:* Grant: Each round-robin arbiter associated with the output link *LC(r, j)* chooses one request by searching from the position of *PC(r, j)*, and sends the grant to the matched *LI(i, r)* of *IM(i)*.
  - *Step 3:* Open Grant: If after step 2, the unmatched output links *LC(r, j)* still exist, each unmatched output link *LC(r, j)* sends the open grant to the output link *LI(i, r)* of the *IM(i)*. The open grant contains the idle output's number of the CM module, which simultaneously determine the *OM(j)* and accessible outputs of the Clos switching fabric.
  - *Step 4:* If the *LI(i, r)* receives the grant from the *LC(r, j)* it sends the cell, at the next time slot, from the matched *VOQ(i, v)* to the *OP(j, h)* through the *CM(r)*. If the *LI(i, r)* receives the open grant from the *LC(r, j)* the open grant arbiter has to choose one cell, which is destined to *OM(j)* and sends it at the next time slot. The open grant arbiter starts to go through the VOQs looking for the proper cell from the position shown by

the *POG(i, k)* pointer. The IM cannot send the cell without receiving the grant or the open grant. Not granted requests will be again attempted to be matched at the next time slot because the pointers are updated only if the matching is achieved. If the cell is sent as a reaction to the open grant the pointers are updated under the following conditions:

- if the pointer *PL(i, r)* points the VOQ which sent the cell, it is updated;
- if the pointer *PV(i, v)* points the output used to sent the cell, it is updated;
- if the pointer *PC(r, j)* points the link *LI(i, r)* used to sent the open grant, it is updated.

Fig. 5-10 illustrates the details of the CRRD-OG algorithm by showing an example for the Clos network *C(3, 3, 3)*.

- *PHASE 1:* Matching within *IM(2)* (one iteration).
  - *Step 1:* The nonempty VOQs: *VOQ(2, 0), VOQ(2, 2), VOQ(2, 3), VOQ(2, 4)*, and *VOQ(2, 8)* send requests to all output link arbiters (Fig. 5).

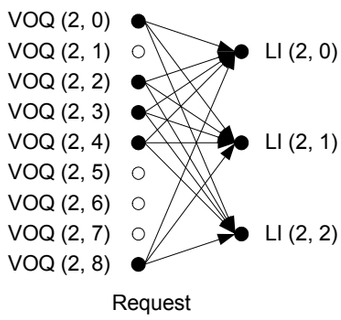

Fig. 5. Nonempty VOQs send requests to all output link arbiters

  - *Step 2:* Output link arbiters associated with *LI(2, 0), LI(2, 1)* and *LI(2, 2)* select *VOQ(2, 0), VOQ(2, 2)* and *VOQ(2, 3)* respectively, according to their pointers position and send grants to them (Fig. 6).

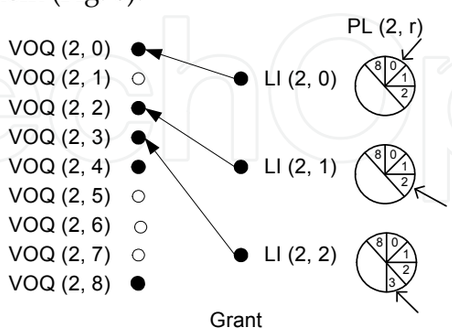

Fig. 6. Output link arbiters send grants to selected VOQs

  - Step 3. Each selected VOQ: *VOQ(2, 0), VOQ(2, 2)* and *VOQ(2, 3)*, receives only one grant, and sends accept to the proper output link arbiter (Fig. 7).

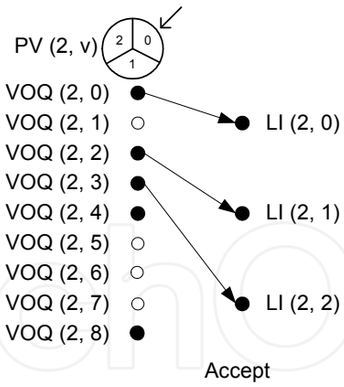

Fig. 7. VOQs send accept to chosen output link arbiters

- PHASE 2: Matching between IM and CM (as an example we consider the state in *CM(2)*).
  - Step 1. In this step the output links of *CM(2)* receive requests from the output links of IMs matched in Phase 1. The requests are as follows: *LC(2, 0), LC(2, 1), LC(2, 0)* (Fig. 8).

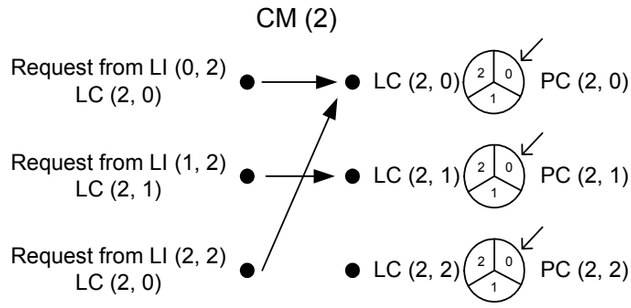

Fig. 8. Output link arbiters of the CM(2) receive requests

  - Step 2. The output link arbiter *LC(2, 0)* receives two requests from *IM(0)* and *IM(2)*, and selects the request from *IM(0)*, according to the pointer position. The output link arbiter *LC(2, 1)* selects request from *IM(2)*. Output links arbiters: *LC(2, 0)* and *LC(2, 1)* send grants to *IM(0)* and *IM(1)* respectively.
  - Step 3. The output link arbiter *LC(2, 2)* does not receive a request, so it sends open grant to *IM(2)* (Fig. 9).

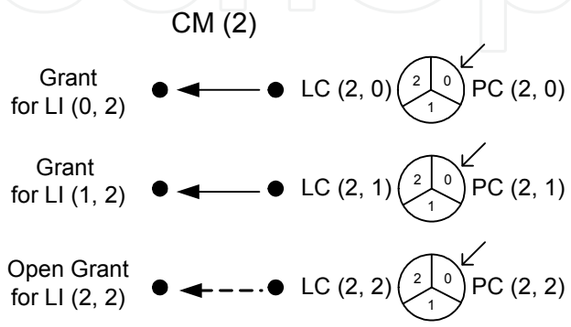

Fig. 9. The output port arbiter LC(2, 2) sends the open grant to LI (2, 2)

- Step 4. *IM(2)* receives the open grant from *LC(2, 2),* which means that it is possible to send one cell to *OP(2, h)*. It chooses a cell from *VOQ(2, 8)*. The cell is destined to *OP(2, 2)* (Fig. 10), and is sent at the next time slot, together with other cells from IMs to OMs through CMs.

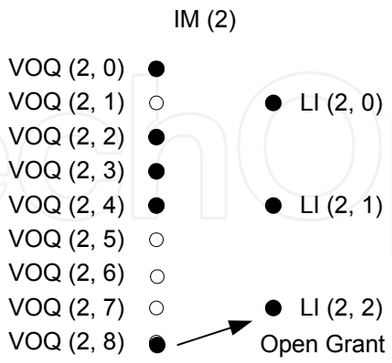

Fig. 10. The cell from VOQ(2, 8) is matched with LI(2, 2), as a reaction to the open grant received from LC(2, 2)

3.4 Concurrent Master-Slave Round-Robin Dispatching Scheme

The Concurrent Master-Slave Round-Robin Dispatching (CMSD) algorithm is an improved version of the CRRD algorithm. It preserves all advantages of the CRRD scheme but more arbiters are used to perform the iterative matching process within the IMs. Two sets of round-robin arbiters (master and slave) are employed to perform hierarchical round-robin arbitration process in the first stage of the Clos-network switch. Each output link of IMs is associated with one master and *k* slaves arbiters. To describe the CMSD algorithm we define several notations based on the terminology used in (Oki at al., 2002a). A VOQ group that consists of *n* VOQs storing cells from *IM(i)* to *OM(j)* is denoted by *G(i, j)*. Each IM has *m* master output-link round robin arbiters, denoted as *ML(i, r), mk* slave output-link round-robin arbiters, denoted as *SL(i, j, r),* and *nk* VOQ round-robin arbiters. Each master arbiter associated with *LI(i, r)* has its own pointer *PML(i, r)*. Each slave arbiter associated with *LI(i, r)* and *G(i, j)* has its own pointer *PSL(i, j, r)*. Each VOQ arbiter associated with *VOQ(i, j, h)* has its own pointer *PV(i, j, h)*. The master arbiter is responsible for selection of one nonempty *G(i, j)* group, while the slave arbiter selects one nonempty VOQ within that VOQ group.

The CMSD algorithm works as follows:

- *PHASE 1:* Matching within IM

  First iteration:

  - *Step 1:* Request: Two sets of requests are sent to the output link arbiters. The group-level request are sent from *G(i, j)* that has at least one non-empty VOQ to every master arbiter *ML(i, r)*. At the same time, each nonempty *VOQ(i, j, h)* sends a request to every slave arbiter *SL(i, j, r)*.
  - *Step 2:* Grant: Each round-robin master arbiter *ML(i, r)* chooses a request by searching from the position of *PML(i, r)*. At the same time, each slave arbiter selects one VOQ request in a round-robin fashion by searching from the position of *PSL(i, j, r)*. The slave arbiter *SL(i, j, r)* will send the grant to the selected VOQ only if *G(i, j)* has been

selected by its master arbiter. If *SL(i, j, r)* does not receive a grant, the search is invalid.

- *Step 3:* Accept: Each VOQ arbiter searches in a round-robin fashion one grant, among all those received, and sends the accept to the master and slave output-link arbiters. Each arbiter starts searching a grant from the position of *PV(i, j, h)*.

i-th iteration (*i>1*):

- *Step 1:* Each *VOQ(i, j, h)* unmatched at the previous iterations sends another request to the slave arbiters. The group *G(i, j)*, which has at least one unmatched nonempty VOQ sends a request to the master arbiters.
- *Step 2 and 3:* These steps are the same as in the first iteration.

➢ *PHASE 2:* Matching between IM and CM – the matching procedure is the same as in the CRRD algorithm.

All the round-robin pointers located in *IM(i)* (namely: *PML(i, r), PSL(i, j, r)* and *PV(i, j, h))* and in *CM(r)* (namely: *PC(r, j))* are updated to one position after the granted position only if the matching is achieved at Phase 1 and the request is also granted at Phase 2. The CMSD algorithm can very easily achieve the desynchronization effect of all round-robin pointers, so it can provide high throughput without expansion under the uniform traffic.

3.5 Static Round-Robin Dispatching

The Static Round-Robin Dispatching (SRRD) algorithm was proposed by K. Pun and M. Hamdi, and is an adaptation of the Static Round-Robin (SRR) scheme for the MSM Clos-network switches. The SRR algorithm was first introduced by Jiang and Hamdi in (Jiang & Hamdi, 2001) for crossbar switches and uses the same handshaking scheme as in the iSlip or DRRM scheme (Chao & Liu, 2007). The algorithm is simple and can achieve very good delay performance. In this algorithm the arbitration pointers are artificially set to be desynchronized at the beginning, and are updated in a static way to keep them desynchronized all the time. Additionally, the grant and accept pointers are "mutual matched". That is, if grant pointer g_j in output j is pointing to input i, then accept pointer a_i in input i must point to output j. The matching sequence in SRR scheme is shown in Fig. 11. This allows the maximum matching from input ports to output ports if all VOQs have a cell to be sent.

The SRRD scheme works in the same way as the CMSD scheme, which means that the phases and steps are in both algorithms identical except the pointer initialization and pointer updating. The initial values of the pointers are as follows: *PV(i, j, h)=h, PSL(i, j, r)=r, PML(i, r)=(i+r)%k* and *PC(r, j)=i if PML(i, r)=j*. The pointers *PML(i, r)* and *PC(r, j)* are always incremented by one (mod k), but the pointers *PV(i, j, h)* and *PSL(i, j, r)* remain unchanged, no matter there is a match or not.

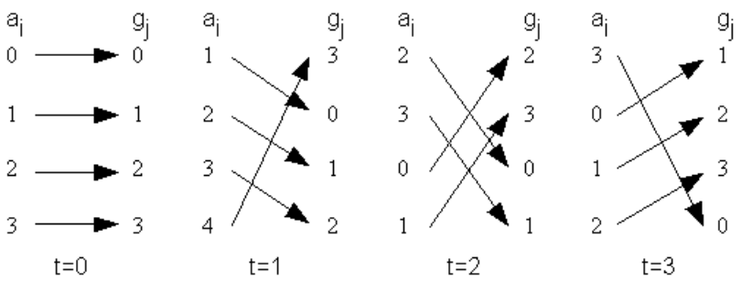

Fig. 11. Matching sequence in SRR algorithm

The SRRD scheme can always achieve 100% throughput under the uniform traffic. Unfortunately, due to several arbiters may grant the same request at the same time, the performance under nonuniform traffic is degraded. This phenomenon appears because all conventional arbiters search in clock-wise direction. To improve the performance of the MSM Clos switch under the nonuniform traffic distribution patterns it is necessary to allow some round-robin arbiters to search the requests in clockwise direction and anti-clockwise direction alternatively, each for one time slot. The 0/1 counter is necessary to keep track of time. The counter is incremented by one (mod 2) in each time slot. If counter shows 0 the master arbiter *ML(i, r)* searches one request in clockwise round-robin fashion, otherwise if counter shows 1, the master arbiter searches one request in anti-clockwise round-robin fashion.

3.6 Performance of CRRD, CMSD, SRRD and CRRD-OG algorithms

A. Packet Arrival Models

Two packet arrival models namely the Bernoulli and bursty are considered in simulation experiments. In the Bernoulli arrival model cells arrive at each input in slot-by-slot manner and the probability that there is a cell arriving in each time slot is identical and independent of any other slot. The probability that a cell may arrive in a time slot is denoted by p and is referred to as the load of the input. This type of traffic defines a memoryless random arrival pattern.

In the bursty traffic model, each input alternates between active and idle periods. During active periods, cells destined for the same output arrive continuously in consecutive time slots. The average burst (active period) length is set to 16 cells in our simulations.

B. Traffic distribution models

We consider several traffic distribution models which determine the probability that a cell which arrives at an input will be directed to a certain output. The considered traffic models are:

Uniform traffic – this type of traffic is the most commonly used traffic profile. In the uniformly distributed traffic probability p_{ij} that a packet from input i will be directed to output j is uniformly distributed through all outputs, i.e.:

$$p_{ij} = p/N \; \forall i, j \tag{1}$$

Trans-diagonal traffic – in this traffic model some outputs have a higher probability of being selected, and respective probability p_{ij} was calculated according to the following equation:

$$p_{ij} = \begin{cases} \frac{p}{2} & for \;\; i = j \\ \frac{p}{2(N-1)} & for \;\; i \neq j \end{cases} \tag{2}$$

Bi-diagonal traffic – is very similar to the trans-diagonal traffic but packets are directed to one of two outputs, and respective probability p_{ij} was calculated according to the following equation:

$$p_{ij} = \begin{cases} \frac{2}{3}p & for\ \ i = j \\ \frac{p}{3} & for\ \ j = (i+1) \mod N \\ 0 & otherwise \end{cases} \tag{3}$$

Chang's traffic – this model is defined as:

$$p_{ij} = \begin{cases} 0 & for\ \ i = j \\ \frac{1}{N-1} & otherwise \end{cases} \tag{4}$$

The experiments have been carried out for the MSM Clos switching fabric of size 64 × 64 - C(8, 8, 8), and for a wide range of traffic load per input port: from p = 0.05 to p = 1, with the step 0.05. The 95% confidence intervals that have been calculated after t-student distribution for ten series, per 55000 cycles each (after the starting phase comprising 15000 cycles, which enables to reach the stable state of the switching fabric), are at least one order lower than the mean value of the simulation results, therefore they are not shown in the figures. We have evaluated two performance measures: the average cell delay in time slots and the maximum VOQs size for the CRRD, CMSD, SRRD, and CRRD-OG algorithms. The results of the simulation under 1 and/or 4 iterations (represented in figures by itr) are shown in the charts (Fig. 12-21). In any case, the number of iterations between any IM and CM is one.

Fig. 12, 14, 16, 18 show the average cell delay in time slots obtained for the uniform, Chang's, trans-diagonal and bi-diagonal traffic patterns, whereas Fig. 13, 15, 17, 19 show the maximum VOQ size in a number of cells. To make the charts more clear and lucid only results for itr=4 are shown in figures concerning the maximum VOQ size. Fig. 20 and 21 show the results for the bursty traffic with the average burst length set to 16 cells.

We can observe that using the Bernoulli traffic and all investigated traffic distribution patterns the CRRD-OG algorithm provides better performance than the CRRD, CMSD and SRRD algorithms. In many cases the CRRD-OG algorithm with one iteration delivers better performance than other algorithms with four iterations (see Fig. 12, 14, 16). The same relation between the CRRD-OG scheme and others schemes we can notice under the bursty traffic (Fig. 20).

Under the uniform traffic the SRRD scheme gives only slightly worse results than the CRRD-OG scheme; the worst result gives pure CRRD algorithm. The same relation we can see in Fig. 13 which shows the comparison of the maximum VOQ size. The biggest buffers we need if we control the MSM Clos-network switch using the CRRD algorithm. The Chang's distribution traffic pattern is very similar to the uniform distribution traffic pattern. Under this traffic distribution pattern all algorithms receive 100% throughput and CRRD-OG scheme with one iteration delivers better performance than other algorithms with four iterations for the cell delay as well as the maximal VOQ size. (Fig. 14, 15). The trans-diagonal and bi-diagonal traffic distribution patterns are highly demanding and the investigated packet dispatching schemes cannot provide the 100% throughput for the MSM Clos – network switch. The best results have been obtained for the CRRD-OG scheme with 4 iterations. These are respectively: under trans-diagonal traffic pattern - 80% throughput for one iteration and 85% throughput for four iterations (Fig. 16) and under bi-diagonal traffic pattern – 95% (Fig. 18). Under the bursty packet arrival model the CRRD-OG scheme

provides much better performance than other algorithms especially for the very high input load (Fig. 20). The same relationship as for the cell delay we can observe for the maximal VOQs size (Fig. 13, 15, 17, 19, 21). It is obvious that for small cell delay the size of VOQs will be also small.

The simulation experiments have shown that the CRRD-OG scheme with one iteration gives very good results in the average cell delay and VOQs size. An increase in the number of iterations do not produce further significant improvement, quite the opposite to other iterative algorithms. Particularly more than n/2 iterations do not change significantly the performance of all investigated iterative schemes.

The investigated packet dispatching schemes are based on the effect of desynchronization of arbitration pointers in the Clos-network switch. In our research we have made an attempt to improve the method of pointers desynchronization for the CRRD-OG scheme, to ensure the 100% throughput for the nonuniform traffic distribution patterns. Additional pointers and arbiters for open grants had been added to the MSM Clos-network switch, but the scheme was not able to provide 100% throughput for the nonuniform traffic distribution patterns. To our best knowledge it is not possible to achieve very good desynchronization of pointers using the methods implemented in the iterative packet dispatching schemes. In our opinion the decisions of the distributed arbiters have to be supported by the central arbiter, but the implementation of such solution in the real equipment will be very complex.

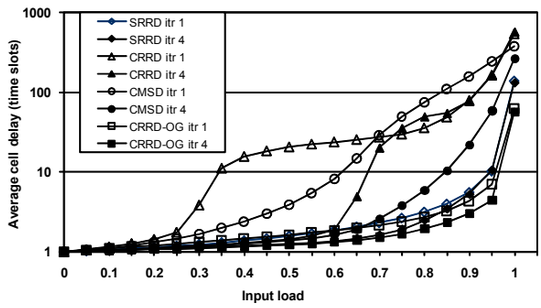

Fig. 12. Average cell delay, uniform traffic

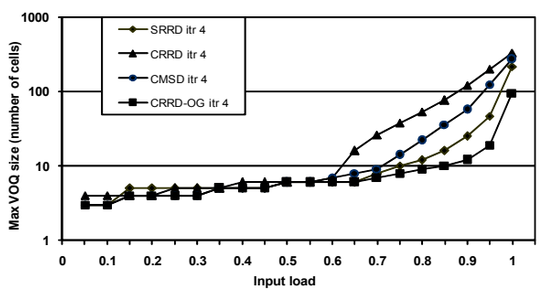

Fig. 13. Maximum VOQ size, uniform traffic

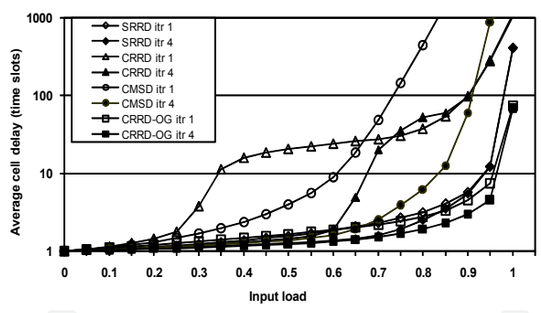

Fig. 14. Average cell delay, Chang's traffic

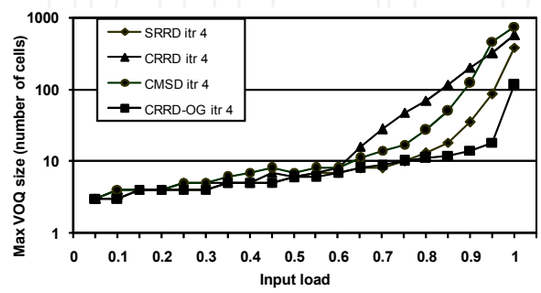

Fig. 15. Maximum VOQ size, Chang's traffic

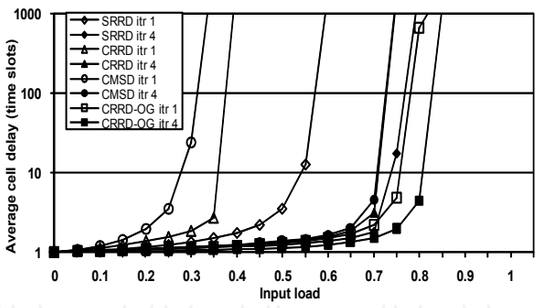

Fig. 16. Average cell delay, trans-diagonal traffic

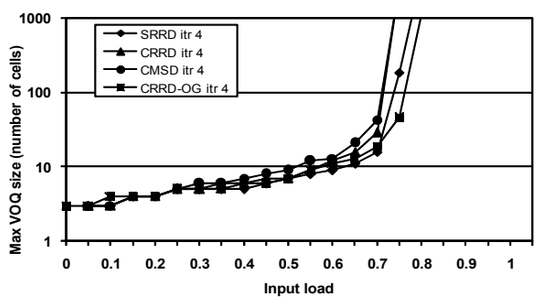

Fig. 17. Maximum VOQ size, trans-diagonal

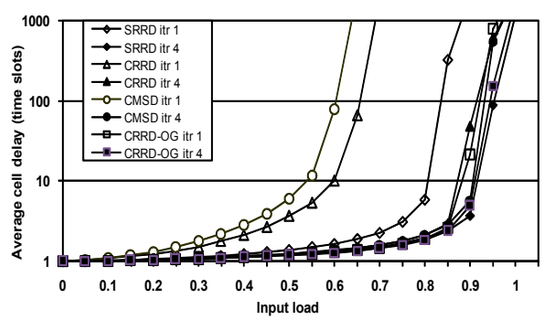

Fig. 18. Average cell delay, bi-diagonal traffic

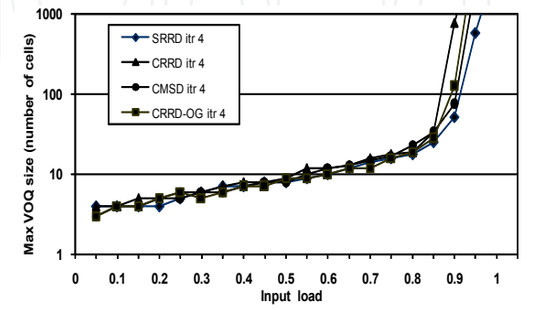

Fig. 19. Maximum VOQ size, bi-diagonal traffic

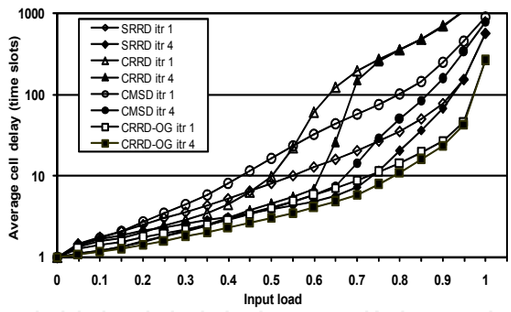

Fig. 20. Average cell delay, bursty traffic, average burst length b=16

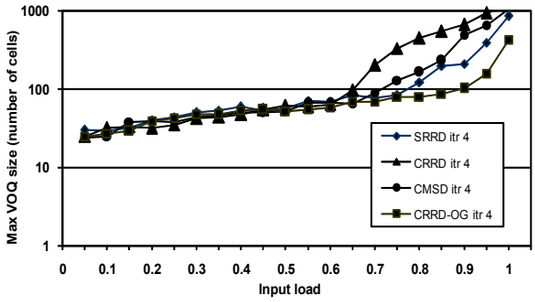

Fig. 21. Maximum VOQ size, bursty traffic, average burst length b=16

4. Packet dispatching algorithms with centralized arbitration

The packet dispatching algorithms with centralized arbitration use a central arbiter to take packet scheduling decisions. Currently, the central arbiters are used to control one-stage switching fabrics. This subchapter presents three packet dispatching schemes with centralized arbitration for the MSM Clos-network switches. We call these schemes as follows: Static Dispatching-First Choice (SD-FC), Static Dispatching-Optimal Choice (SD-OC) and Input Module - Output Module Matching (IOM).

Packet switching nodes in the next generation Internet should be ready to support the nonuniform/hot spot traffic. Such case often occurs when a popular server is connected to a single switch/router port. Under the nonuniform traffic distribution patterns selected VOQs store more cells than others. Due to some input buffers may be overloaded, it is necessary to implement to a packet dispatching scheme a special mechanism, which is able to send up to n cells from *IM(i)* to *OM(j)* in the same time slot, in order to unload overloaded buffers. Three dispatching schemes presented in this subchapter have such possibility.

The SD-FC, SD-OC, and IOM schemes make a matching between each IM and OM, taking into account the number of cells waiting in VOMQs. Each VOMQ has its own counter *PV(i, j)*, which shows the number of cells destined to *OM(j)*. The value of *PV(i, j)* is increased by 1 when a new cell is written into a memory, and decreased by 1 when a cell is sent out to *OM(j)*. The algorithms use the central arbiter to indicate the matched pairs of *IM(i)*-*OM(j)*. The set of data sent to the arbiter by each scheme is different, therefore, the architecture and functionality of each arbiter is also different. After a matching phase, in the next time slot *IM(i)* is allowed to send up to n cells to the selected *OM(j)*.

In the SD-OC and SD-FC schemes the central arbiter matches *IM(i)* and *OM(j)* only if the number of cells buffered in *VOMQ(i, j)* is at least equal to n. Under the nonuniform traffic distribution patterns it happens very often, contrary to the uniform traffic distribution. In the proposed packet dispatching schemes each VOMQ has to wait until at least n cells are stored before being allowed to make a request. In simulation experiments we consider the Clos switching fabric without any expansion, denoted by *C(n, n, n)*, so in description of the packet dispatching schemes, k and m parameters are not used.

4.1 Static Dispatching

To reduce latency and avoid starvation, a very simple packet dispatching routine, called Static Dispatching (SD), is also used in the MSM Clos-network switch to support SD-FC and SD-OC schemes. Under this algorithm, connecting paths in switching fabric are set up according to static, but different in each CM, connection patterns (see Fig. 22). These fixed connection paths between IMs and OMs eliminate the handshaking process with the second stage, and no internal conflicts in the switching fabric will occur. Also no arbitration process is necessary. Cells destined to the same OM, but located in different IMs, will be sent through different CMs.

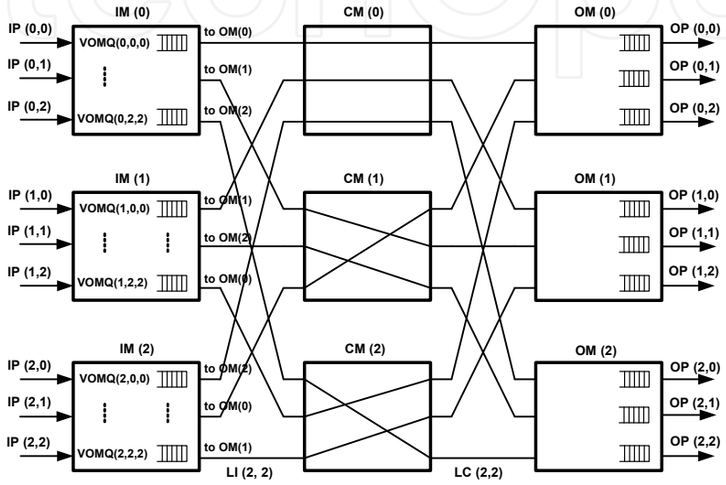

Fig. 22. Static connection patterns in CMs, *C(3, 3, 3)*.

In detail, the SD algorithm works as follows:

- *Step 1:* According to the connection pattern of *IM(i)*, match all output links *LI(i, r)* with cells from VOMQs.
- *Step 2:* Send the matched cells in the next time slot. If there is any unmatched output link, it remains idle.

4.2 Static Dispatching-First Choice and Static Dispatching-Optimal Choice Schemes

The SD-OC and SD-FC schemes are very similar, but the central arbiter matching IMs and OMs works in a different way. In both algorithms the *PV(i, j)* counter, which reaches the value equal or greater than *n* sends the information about an overloaded buffer to the central arbiter. In the central arbiter there is a binary matrix representing VOMQs load. If the value of matrix element *x[i, j]*=1, it means that *IM(i)* has at least *n* cells that should be sent to *OM(j)*.

In the SD-OC scheme the main task of the central arbiter is to find an optimal set of 1s in the matrix. The best case is *n* 1s, but it is possible to choose only single 1 from column *i* and row *j*. If there is no such set of 1s the arbiter tries to find a set of *n-1* 1s, which fulfills the same conditions, and so on. The round-robin routine is used for the starting point of the searching process. Otherwise, the MSM Clos switching fabric is working under the SD scheme.

The main difference between the SD-OC and SD-FC lies in the operation of the central arbiter. In the SD-FC scheme the central arbiter does not look for the optimal set of 1s, but

tries to match *IM(i)* with *OM(j)*, choosing the first 1 found in column *i* and row *j*. No optimization process for selecting IM-OM pairs is employed. In detail, the SD-OC algorithm works as follows:

- *Step 1*: *(each IM)*: If the value of *PV(i, j)* counter is equal to or greater than *n*, send a request to the central arbiter.
- *Step 2*: *(central arbiter)*: If the central arbiter receives the request from *IM(i)*, it sets the value of the buffer load matrix element *x[i, j]* to 1 (the values of *i* and *j* come from the counter *PV(i, j)*).
- *Step 3*: *(central arbiter)*: After receiving all requests, the central arbiter tries to find an optimal set of 1s, which allows to send the most number of cells from IMs to OMs. The central arbiter has to go through all rows of the buffer load matrix to find a set of *n* 1s representing *IM(i)* and *OM(j)* matching. If there is not possible to find a set of *n* 1s it attempts to find a set of *(n-1)* 1s, and so on.
- *Step 4*: *(each IM)*: In the next time slot send *n* cells from IMs to the matched OMs. Decrease the value of *PV(i, j)* by *n*. For IM-OM pairs not matched by the central arbiter use the SD scheme and decrease the value of *PV* counters by 1.

The steps in the SD-FC scheme are the same as in the SD-OC scheme, but the optimization process in the third step is not carried out. The central arbiter chooses the first 1, which fulfill the requirements in each row. The row searched as the first one is selected according to the round robin routine.

4.3 Input-Output Module matching algorithm

The IOM packet dispatching scheme employs also the central arbiter to make a matching between each IM and OM. The cells are sent only between IM-OM pairs matched by the arbiter. The SD scheme is not used.

In detail, the IOM algorithm works as follows:

- *Step 1: (each IM)*: Sort the values of *PV(i, j)* in descending order. Send to the central arbiter a request containing a list of the OMs identifiers. The identifier of *OM(j)* to which *VOMQ(i, j)* stores the most number of cells should be placed on the list as the first one, and the identifier of *OM(s)* to which *VOMQ(i, s)* stores the least number of cells should be placed on the list as the last one.
- *Step 2: (central arbiter)*: The central arbiter analyzes one by one the requests received from IMs and checks if it is possible to match *IM(i)* with *OM(j)*, the identifier of which was sent as the first one on the list in the request. If the matching is not possible, because the *OM(j)* is matched with other IM, the arbiter selects the next OM on the list. The round-robin arbitration is employed for selection of *IM(i*) the request of which is analyzed as the first one.
- *Step 3: (central arbiter)*: The central arbiter sends to each IM confirmation with the identifier of *OM(t)*, to which the IM is allowed to send cells.
- *Step 4: (each IM)*: Match all output links *LI(i, r)* with cells from *VOMQ(i, t)*. If there is less than *n* cells to be sent to *OM(t)*, some output links remain unmatched.
- *Step 5: (each IM)*: Decrease the value of *PV(i, t)* by the number of cells which will be sent to *OM(t)*.
- *Step 6: (each IM)*: In the next time slot send the cells from the matched *VOMQ(i, t)* to the *OM(t)* selected by the central arbiter.

4.4 Performance of SD-FC, FD-OC and IOM schemes

The simulation experiments were carried out under the same conditions as the experiments for the distributed arbitration (see subchapter 3.6). We have evaluated two performance measures: average cell delay in time slots and maximum VOMQs size (we have investigated the worst case). The size of the buffers at the input and output side of switching fabric is not limited, so cells are not discarded. However, they encounter the delay instead. Because of the unlimited size of buffers, no mechanism controlling flow control between the IMs and OMs (to avoid buffer overflows) is implemented. The results of the simulation for the Bernoulli arrival model are shown in the charts (Fig. 23-32). Fig. 23, 25, 27, 29 show the average cell delay in time slots obtained for the uniform, Chang's, trans-diagonal, bi-diagonal, and bursty traffic patterns, whereas Fig. 24, 26, 28, 30 show the maximum VOMQ size in number of cells. Fig. 31, 32 show the results for the bursty traffic with the average burst size b=16, and uniform traffic distribution pattern.

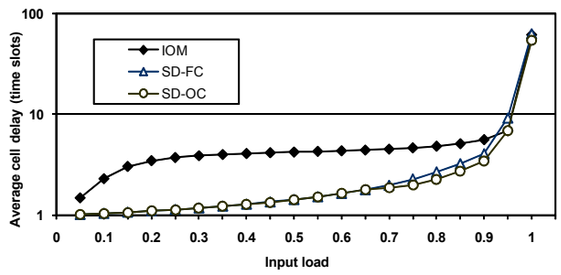

Fig. 23. Average cell delay, uniform traffic

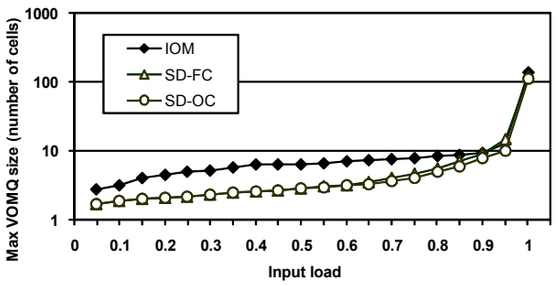

Fig. 26. The maximum VOMQ size, Chang's traffic

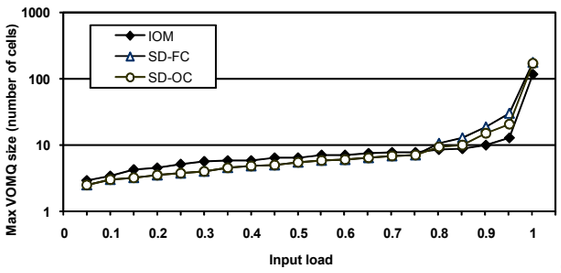

Fig. 24 The maximum VOMQ size, uniform traffic

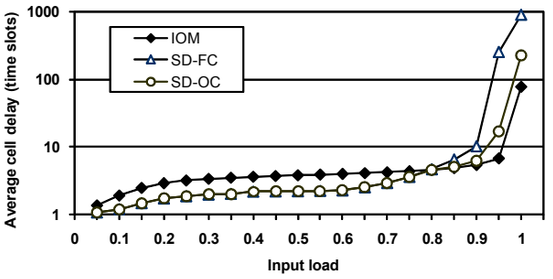

Fig. 27. Average cell delay, trans-diagonal traffic

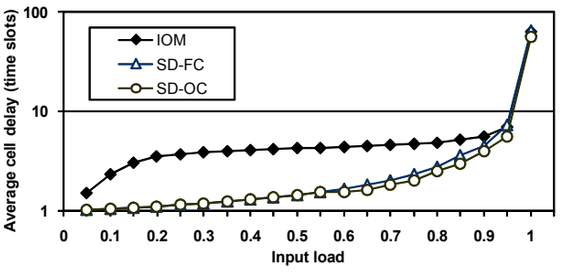

Fig. 25. Average cell delay, Chang's traffic

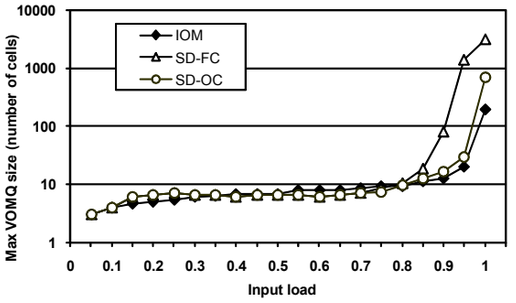

Fig. 28 The maximum VOMQ size, trans-diagonal traffic

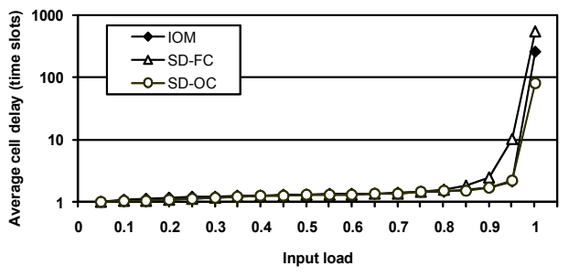

Fig. 29. Average cell delay, bi-diagonal traffic

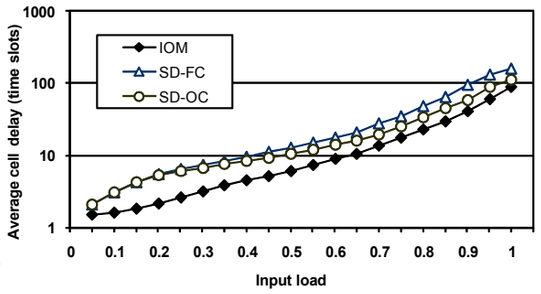

Fig. 31. Average cell delay, bursty traffic

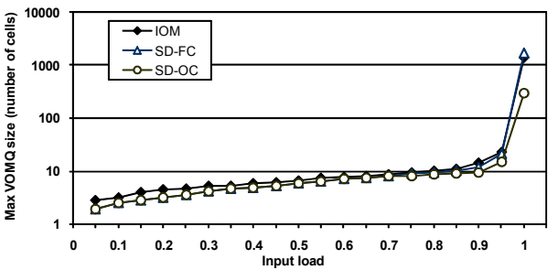

Fig. 30. The maximum VOMQ size, bi-diagonal traffic

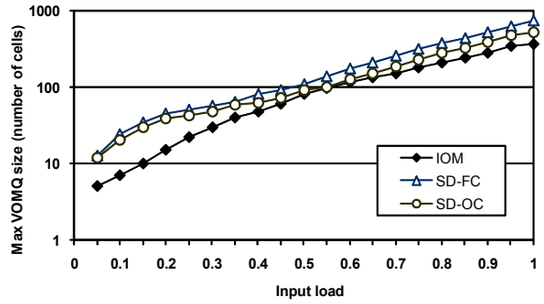

Fig. 32. The maximum VOMQ size, bursty traffic

We can see that the MSM Clos-network switch with all the schemes proposed achieves 100% throughput for all kinds of investigated traffic distribution patterns under Bernoulli arrival model and for the bursty traffic. The average cell delay is less than 10 for wide range of input load, regardless of the traffic distribution pattern. It is a very interesting result especially for the trans-diagonal and bi-diagonal traffic patterns. Both traffic patterns are highly demanding and many packet dispatching schemes proposed in the literature cannot provide the 100% throughput for the investigated switching fabric. For the bursty traffic, the average cell delay grows very similar to linear function of input load with the maximum value less than 150. We can see that the very complicated arbitration routine used in the SD-OC scheme does not improve the performance of the MSM Clos-network switch. In some cases the results are even worse than for IOM scheme (the trans-diagonal traffic with very high input load and the bursty traffic – Fig. 27 and 31). Generally, the IOM scheme gives higher latency than the SD schemes, especially for low to medium input load. It is due to matching *IM(i)* to that *OM(j)* to which it is possible to send the most number of cells. As a consequence, it is less probable to match IM-OM pairs to serve one or two cells per cycle.

The size of VOMQ in the MSM Clos switching network depends on the traffic distribution pattern. For all presented packet distribution schemes and the uniform and Chang's traffic the maximum size of VOMQ is less than 140 cells. It means that in the worst case, the average number of cell waiting for transmission to particular output was not bigger than 16. For the trans-diagonal traffic and the IOM scheme the maximum size of VOMQ is less than 200, but for the SD-OC and SD-FC the size is greater and come to 700 and 3000 respectively. For the bi-diagonal traffic the smallest size of VOMQ was obtained for the SD-OC scheme -

less than 290. For the bursty traffic the maximal size of VOMQ comes to: 750 for the SD-FC, 500 for the SD-OC and 350 for the IOM scheme.

5. Related Works

The field of packet scheduling in VOQ switches boasts of an extensive literature. Many algorithms are applicable to the single-stage (crossbar) switches and are not useful for packet dispatching in the MSM Clos-network switches. Some of them are more oriented to implementation, whereas others are of more theoretical significance. Here we review a representation of the works concerning packet dispatching in the MSM Clos-network switches.

Pipeline-Based Concurrent Round Robin Dispatching
E. Oki at al. have proposed in (Oki at al., 2002b) the Pipeline-Based Concurrent Round Robin Dispatching (PCRRD) scheme for the Clos-network switches. The algorithm can relax the strict timing constraint required by the CRRD and CMSD schemes. These algorithms have constrained dispatching scheduling to one cell slot. The constraint is a bottleneck when the switch capacity increases. The PCRRD scheme is able to relax the scheduling time into more than one time slot, however nk^2 request counters and P subschedulers have to be used to support the dispatching algorithm. Each subscheduler is allowed to take more than one time slot for packet scheduling, whereas one of them provides the dispatching result every time slot. The subschedulers adopt the CRRD algorithm, but other schemes (like CMSD) may be also adopted. Both, the centralized and non-centralized implementations of the algorithm are possible. In the centralized approach, each subscheduler is connected to all IMs. In the non-centralized approach, the subschedulers are implemented in different locations i.e. in IMs and CMs. The PCRRD algorithm provides 100% throughput under uniform traffic and ensures that cells from the same VOQ are transmitted in sequence.
Maximum Weight Matching Dispatching
The Maximum Weight Matching Dispatching scheme (MWMD) for the MSM Clos-network switches was proposed by R. Rojas-Cessa at al. in (Rojas-Cassa at al., 2004). The scheme is based on the maximum weight matching algorithm implemented in input-buffered single-stage switches. To perform the MWMD scheme each *IM(i)* has k virtual output-module queues (VOMQs) to eliminate HOL blocking. VOMQs are used instead of VOQs and *VOMQ(i, j)* stores cells at *IM(i)* destined to *OM(j)*. Each VOMQ is associated with m request queues (RQ), each denoted as *RQ(i, j, r)*. The request queue *RQ(i, j, r)* is located in *IM(i)* and stores requests of cells destined for *OM(j)* through *CM(r)* and keeps the waiting time *W(i, j,r)*. The waiting time represents the number of slots a head-of-line request has been waiting. When a cell enters *VOMQ(i, j)*, the request is randomly distributed and stored in *RQ(i, j, r)* among m request queues. A request in *RQ(i, j, r)* is not related to a specific cell but to *VOMQ(i, j)*. A cell is sent from *VOMQ(i, j)* to *OM(j)* in a FIFO manner when a request in *RQ(i, j, r)* is granted.
The MWMD scheme uses a central scheduler which consists of m subschedulers, denoted as *S(r)*. Each subscheduler is responsible for selecting requests related to cells which can be transmitted through *CM(r)* at the next time slot e.g.: subscheduler *S(0)* selects up to k requests from k^2 RQs, where corresponding cells to the selected RQs are transmitted through *CM(0)* at the next time slot. *S(r)* selects one request from each IM and one request to each OM according to the Oldest-Cell-First (OCF) algorithm. The OCF algorithm uses the waiting

time *W(i, j, r)* which is kept by each *RQ(i, j, r)* queue. *S(r)* finds a match *M(r)* at each time slot, so that the sum of *W(i, j, r)* for all *i* and *j*, and a particular *r* is maximized. It should be stressed that each subscheduler behaves independently and concurrently, and uses only k^2 *W(i, j, r)* to find *M(r)*.

When *RQ(i, j, r)* is granted by *S(r)*, the HOL request in *RQ(i, j, r)* is dequeued and a cell from *VOMQ(i, j)* is sent at the next time slot. The cell is one of the HOL cells in *VOMQ(i, j)*. The number of cells sent to OMs is equal to the number of granted requests by all subschedulers.

R. Cessa at al. has proved that the MWMD algorithm achieves 100% throughput for all admissible independent arrival processes without internal bandwidth expansion, i.e. *n=m* for the Clos MSM network.

Maximal Oldest Cell First Matching Dispatching

The Maximal Oldest-cell first Matching Dispatching (MOMD) scheme was proposed by R. Rojas-Cessa at al. in (Rojas-Cassa at al., 2004). The algorithm has lower complexity for a practical implementation than MWMD scheme. The MOMD scheme uses the same queues arrangement as MWMD scheme: *k* VOMQs at each IM, each denoted as *VOMQ(i, j)* and *m* request queues, RQs, each associated with a VOMQ, each denoted as *RQ(i, j, r)*. Each cell enters a *VOMQ(i, j)* gets a time stamp. A request with the time stamp is stored in *RQ(i, j, r)*, where *r* is randomly selected. The distribution of the requests can also be done in the round-robin fashion among RQs. The MOMD uses distributed arbiters in IMs and CMs. In each IM, there are *m* output-link arbiters, and in each CM there are *k* arbiters, each of which corresponds to a particular OM. To determine the matching between *VOMQ(i, j)* and the output link *LI(i, r)* each non-empty *RQ(i, j, r)* sends a request to the unmatched output link arbiter associated to *LI(i, r)*. The request includes the time stamp of the associated cell waiting at the HOL to be sent. Each output-link arbiter chooses one request by selecting the oldest time stamp, and sends the grant to the selected RQ and VOMQ. Then, each *LI(i, r)* sends the request to the *CM(r)* belonging to the selected VOMQ. Each round-robin arbiter associated with *OM(j)* grants one request with the oldest time stamp and sends the grant to *LI(i, r)* of *IM(i)*. If an IM receives a grant from a CM, the IM sends a HOL cell from that VOMQ at the next time slot. There is possible to consider more iteration between IM and CM within the time slot.

The delay and throughput performance of 64×64 Clos-network switch, where *n=m=k=8* under MOMD scheme are presented in (Rojas-Cassa at al., 2004). The scheme cannot achieve the 100% throughput under uniform traffic with a single IM-CM iteration. The simulation shows that CRRD scheme is more effective under uniform traffic than the MOMD, as the CRRD achieves high throughput with one iteration. However, as the number of IM-CM iterations increases, the MOMD scheme gets higher throughput e.g. in the switch under simulation, the number of iterations to provide 100% throughput is four. The MOMD scheme can provide high throughput under a nonuniform traffic pattern (opposite to the CRRD scheme), called unbalanced, but the number of IM-CM iterations has to be increased to eight. The unbalanced traffic pattern has one fraction of traffic with uniform distribution and the other faction *w* of traffic destined to the output with the same index number as the input; when *w=0*, the traffic is uniform; when *w=1* the traffic is totally directional.

Frame Occupancy-Based Random Dispatching and Frame Occupancy-Based Concurrent Round-Robin Dispatching

The Frame occupancy-based Random Dispatching (FRD) and Frame occupancy-based Concurrent Round-Robin Dispatching (FCRRD) schemes were proposed by C-B. Lin and R. Rojas-Cessa in (Lin & Rojas-Cessa, 2005). Frame based scheduling with fixed-size frames was first introduced to improve switching performance in one-stage input-queued switches. C-B. Lin and R. Rojas-Cessa adopted captured-frame concept for the MSM Clos-network switches using RD and CRRD schemes as the basic dispatching algorithms. The frame concept is related to a VOQ and means the set of one or more cells in a VOQ that are eligible for dispatching. Only the HOL cell of the VOQ is eligible per time slot. The captured fame size is equal to the cell occupancy at *VOQ(i, j, l)* at the time t_c of matching the last cell of the frame associated to *VOQ(i, j, l)*. Cells arriving to *VOQ(i, j, l)* at time t_d, where $t_d > t_c$, are considered for matching if a new frame is captured. Each VOQ has a captured-frame size counter denoted as $CF_{i,j,l}(t)$. The value of this counter indicates the frame size at time slot *t*. The $CF_{i,j,l}(t)$ counter takes a new value when the last cell of the current frame of *VOQ(i, j, l)* is matched. Within the FCRRD scheme the arbitration process includes two phases and the request-grant-accept approach is implemented. The achieved match is kept during the frame duration.

The FRD and FCRRD schemes show higher performance under uniform and several nonuniform traffic patterns, as compared to the RD and CRRD algorithms. What's more the FCRRD scheme with two iterations is sufficient to achieve a high switching performance. The hardware and timing complexity of the FCRRD is comparable to that of the CRRD.

Maximal Matching Static Desynchronization Algorithm

The Maximal Matching Static Desynchronization algorithm (MMSD) was proposed by J. Kleban and H. Santos in (Kleban & Santos, 2007). The MMSD scheme uses the distributed arbitration with the request-grant-accept handshaking approach but minimizes the number of iterations to one. The key idea of the MMSD scheme is static desynchronization of arbitration pointers. To avoid collisions in the second stage, all IMs use connection patterns that are static but different in each IM; it forces cells destined to the same OM, but located in different IMs, to be sent through other CMs. In the MMSD scheme two phases are considered for dispatching from the first to the second stage. In the first phase each IM selects up to m VOMQs and assigns them to IM output links. In the second phase requests associated with output links are sent from IM to CM. The arbitration results are sent from CMs to IMs, so the matching between IMs and CMs can be completed. If there is more than one request for the same output link in a CM, a request is granted from this IM which should use a given CM for connection to an appropriate OM, according to the fixed IM connection pattern. If requests come from other IMs, CM grants one request randomly. In each *IM(i)* there is one group pointer *PG(i, h)* and one *PV(i, v)* pointer, where $0 \le v \le nk - 1$. In *CM(r)*, there are *k* round robin arbiters, and each of them corresponds to *LC(r, j)* – an output link to the *OM(j)* – and has its own pointer *PC(r, j)*.

The performance results obtained for the MMSD algorithm are better or comparable with results obtained for other algorithms, but the scheme is less hardware-demanding and seems to be implementable with the current technology in the three-stage Clos-network switches.

The modified MSM Clos switching fabric with SDRUB packet dispatching scheme

The modified MSM Clos switching fabric and a very simple packet dispatching scheme, called Static Dispatching with Rapid Unload of Buffers (SDRUB) were proposed by J. Kleban at al. in (Kleban at al., 2007). The main idea of modification of the MSM Clos switching fabric lies in connecting bufferless CMs to the two-stage buffered switching fabric so as to give the possibility of rapid unload of VOMQs. In this way an expansion in IMs and OMs is used. The maximum number of connected CMs is equal to *m-1*, but it is possible to use less CMs. In practice, the number of CMs significantly influences the performance of the switching fabric. The number of CMs depends on the traffic distribution pattern to be served. Contrary to the MSM Clos switching fabric, in the modified architecture, at each time slot, it is possible to send one cell from each IM to each OM due to direct connecting path between IMs and OMs. The arbitration is necessary for rapid unload of buffers only.

In the SDRUB scheme each VOMQ has its own counter *PV(i, r)* which shows the number of cells destined to *OM(r)*. The SDRUB algorithm uses a central arbiter to indicate the IMs which are allowed to send cells through CMs. Assume that there is *(y-1)* CMs in the modified MSM Clos switching fabric. When *PV(i, r)* reaches the value equal or greater than *y*, it sends the information about the overloaded buffer to the central arbiter. In the central arbiter there is a binary matrix of buffers load. If the value of matrix element *x[i, j]* is 1, it means that *IM(i)* can send *y* cells to *OM(j)*, one through the direct connection and *y-1* through CMs. The central arbiter changes the value of element *x[i, j]* from 0 to 1 only if it is the first 1 in column *i* and row *j*. In other cases the request is rejected. The OM to which *IM(i)* sends cells using CMs is selected according to the round robin routine. No other optimization process for selecting IM-OM pairs for buffers rapid unload is employed.

Simulation experiments have shown that the modified MSM Clos switching fabric achieves very good performance under uniform as well as nonuniform traffic distribution patterns. To manage the trans-diagonal traffic effectively, it is necessary to implement at least *n/2* CMs. For such number of CMs the switching fabric achieves 100% throughput but any smaller number of CMs reduces the throughput of the switching fabric. Under the bi-diagonal traffic the SDRUB algorithm can achieve 100% throughput only when the maximum number of CMs is used. It is obvious that when the number of CMs increases, the throughput increases proportionally. For the uniform traffic pattern the SDRUB scheme gives very good results for one CM.

6. References

Chao, H. J., & Liu, B. (2007). *High Performance Switches and Routers*, John Wiley & Sons, Inc., ISBN: 978-0-470-05367-6, New Jersey

Chao, H. J., Cheuk, H. L. & Oki, E. (2001). *Broadband Packet Switching Technologies: A Practical Guide to ATM Switches and IP Routers,* John Wiley & Sons, Inc., ISBN: 0-471-00454-5, New York

Clos, C. (1953). A Study of Non-Blocking Switching Networks, *Bell Sys. Tech. Jour.*, Vol. 32, pp. 406-424

Hui, J. Y. & Arthurs, E. (1987). A Broadband Packet Switch for Integrated Transport, *IEEE J. Sel. Areas Commun.*, Vol. 5, No. 8, pp. 1264-1273

Jiang, Y. & Hamdi, M. (2001). A fully desynchronized round-robin matching scheduler for a VOQ packet switch architecture", *Proceedings of IEEE High Performance Switching and Routing 2001 – HPSR 2001,* pp. 407–411, US, Texas, Irving

Kabacinski, W. (2005). *Nonblocking Electronic and Photonic Switching Fabrics*, Springer, ISBN: 978-0-387-25431-9

Kleban, J. & Santos, H. (2007). Packet Dispatching Algorithms with the Static Connection Patterns Scheme for Three-Stage Buffered Clos-Network Switches, P*roceedings of IEEE International Conference on Communications 2007 - ICC-2007*, Scotland, Glasgow

Kleban, J. & Wieczorek, A. (2006). CRRD-OG - A Packet Dispatching Algorithm with Open Grants for Three-Stage Buffered Clos-Network Switches, P*roceedings of High Performance Switching and Routing 2006 – HPSR 2006*, pp. 315-320, Poland, Poznan

Kleban, J., Sobieraj, M. & Węclewski, S. (2007). The Modified MSM Clos Switching Fabric with Efficient Packet Dispatching Scheme", *Proceedings of IEEE High Performance Switching and Routing 2007 – HPSR 2007*, US, New York

Lin, C-B & Rojas-Cessa, R. (2005). Frame Occupancy-Based Dispatching Schemes for Buffered Three-stage Clos-Network switches", P*roceedings of 13th IEEE International Conference on Networks 2005,* Vol. 2, pp. 771-775.

McKeown, N., Mekkittikul, A., Anantharam, V. & Walrand, J. (1999), Achieving 100% Throughput in an Input-queued Switch, *IEEE Trans. Commun.,* Vol. 47, Issue 8, pp. 1260-1267

Oki, E., Jing, Z. & Rojas-Cessa, R. & Chao H. J. (2002a). Concurrent Round-Robin-Based Dispatching Schemes for Clos-Network Switches, *IEEE/ACM Trans. on Networking*, Vol. 10, No.6, pp. 830-844

Oki, E., Rojas-Cessa, R. & Chao, H. J. (2002b). PCRRD: A Pipeline-Based Concurrent Round-Robin Dispatching Scheme for Clos-Network Switches, P*roceedings of IEEE International Conference on Communications 2002 - ICC-2002*, pp. 2121-2125, US, New York

Pun, K., & Hamdi, M. (2004). Dispatching schemes for Clos-network switches, *Computer Networks* No. 44, pp.667-679

Rojas-Cessa, R. Oki, E. & Chao, H. J. (2004). Maximum Weight Matching Dispatching Scheme in Buffered Clos-Network Packet Switches, *Proceedings of IEEE International Conference on Communications 2004 - ICC-2004,* pp. 1075-1079, France, Paris

Yoshigoe, K. & Christensen, K. J. (2003). An evolution to crossbar switches with virtual ouptut queuing and buffered cross points, *IEEE Network*, Vol. 17, No. 5, pp. 48-56

Switched Systems

Edited by Janusz Kleban

ISBN 978-953-307-018-6
Hard cover, 174 pages
Publisher InTech
Published online 01, December, 2009
Published in print edition December, 2009

This book presents selected issues related to switched systems, including practical examples of such systems. This book is intended for people interested in switched systems, especially researchers and engineers. Graduate and undergraduate students in the area of switched systems can find this book useful to broaden their knowledge concerning control and switching systems.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Janusz Kleban (2009). Packet Dispatching Schemes for Three-Stage Buffered Clos-Network Switches, Switched Systems, Janusz Kleban (Ed.), ISBN: 978-953-307-018-6, InTech, Available from: http://www.intechopen.com/books/switched-systems/packet-dispatching-schemes-for-three-stage-buffered-clos-network-switches

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen