Rate and Delay Guarantees Provided by Clos Packet Switches with Load Balancing

Aleksandra Smiljanić, *Member, IEEE*

***Abstract*— The size of a single-hop cross-bar fabric is still limited by the technology, and the fabrics available on the market do not exceed the terabit capacity. A multihop fabric such as Clos network provides the higher capacity by using the smaller switching elements (SE). When the traffic load is balanced over the switches in a middle stage, all the traffic would get through the fabric, as long as the switch outputs are not overloaded. However, the delay that packets experience through the Clos switch depends on the granularity of flows that are balanced. We examine the maximum fabric utilization under which a tolerable delay is provided for various load balancing algorithms, and derive the general formula for this utilization in terms of the number of flows that are balanced. We show that the algorithms which balance flows with sufficiently coarse granularity provide both high fabric utilization and delay guarantees to the most sensitive applications. Since no admission control should be performed within the switch, the fast traffic-pattern changes can be accommodated in the proposed scalable architecture.**

I. Introduction

Clos circuit switch has been proposed by Clos in 1953s at Bell Labs [6]. Figure 1 shows the connections between switching elements (SE) in a symmetric Clos three-stage switch. This interconnection rule is: the xth SE in some switching stage is connected to the xth input of each SE in the next stage [6], [7], [8]. Here, all connections have the same bandwidths. It has been shown that a circuit can be established through the Clos switching fabric without rearranging existing circuits as long as the number of SEs in the second stage is at least twice the number of inputs of an SE in the first stage minus 1, i.e. $l \geq 2 \cdot n - 1$. It has also been shown that a circuit can be established through the Clos switching fabric as long as the number of SEs in the second stage is no less than the number of inputs of an SE in the first stage, i.e. $l \geq n$. In the latter case, the number of required SEs and their total capacity are smaller due to the fact that the existing circuits can be rearranged. While the complexity of the switching fabric hardware is reduced, the complexity of the algorithm for a circuit setup is increased. In both cases, non-blocking property of the Clos architecture has been proven assuming the specific algorithms for circuit setup [8]. Various implications of Clos findings have been examined in [12].

The Clos switching fabric can be used for increasing capacity of packet switches as well. The interconnection of SEs would be the same as in the circuit switch case. However, these SEs should be reconfigured in each cell time slot based on the outputs of outstanding cells. Here, packets are split into cells of a fixed duration, which is typically 50ns (64 bytes at 10Gb/s). Algorithms for circuit setup in Clos circuit switches cannot be readily applied in Clos packet switches. First, all SEs should be synchronized on a cell-by-cell basis. Then, an implementation of the algorithm that rearranges connections on a cell-by-cell basis in SEs of a rearrangeable non-blocking Clos switch would be prohibitively complex [7]. So, the Clos fabric with the larger hardware, $l = 2 \cdot n$, is needed for a non-blocking packet switch. A scheduling algorithm that would provide non-blocking in a Clos packet switch would require the higher processing complexity than its counterpart designed for a cross-bar switch [15], [16]. Few heuristics have been proposed to configure SEs in Clos packet switches without assessment of their blocking nature [11], [14].

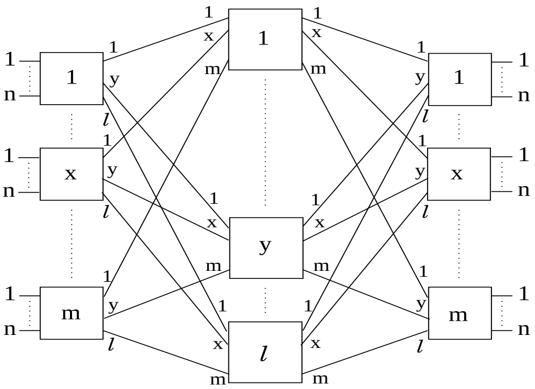
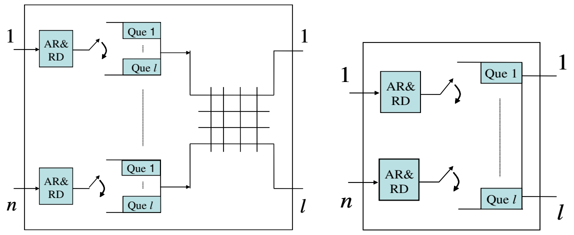


Fig. 1. Clos switching fabric



(a) (b)

Fig. 2. (a) Switching element (SE) based on a cross-bar (b) Switching element based on a shared buffer

On the other side, it has been recognized that a Clos packet switch in which the traffic load is balanced across the SEs provides non-blocking, i.e. with sufficiently large buffers it passes all the traffic if the outputs are not overloaded. Such architecture has been described in [2], [27]. There is a buffering in each stage of the architecture, and the SEs in the heading stages are balancing packets over the SEs in the succeeding stages. Turner showed that the architecture is non-blocking if the traffic of each end-to-end session is balanced over the SEs in a Benes packet switch [27]. We prove in a similar way that a three-stage Clos packet switch based on load balancing is also non-blocking. We focus on the three-stage architecture because it incurs a lower delay than the recursive Benes architecture with the larger number of stages. Advantages of the Clos packet switches with load balancing are multifold. First, their implementation is simple: there is no need for the high-capacity shared buffers or cross-bars, there is no need for the cell-by-cell synchronization across the fabric, and there is no need for the centralized scheduler. Second, the non-blocking property of these switches is an attractive feature: it simplifies network design because the traffic passes the fabric as long as the output ports are not overloaded, and it enables distributed admission control which can follow the fast traffic-pattern changes typical on the Internet. Namely, because the switching fabric is non-blocking, when reserving the bandwidth, an input port only has to check if the output port has enough capacity, or a user has to check if its destination user has enough capacity to receive data. In this paper, we will assess the delay guarantees that can be provided in Clos packet switches.

Load balancing has been proposed in other architectures such as parallel plane switches (PPS) or Birkhoff-von-Neumann switches which are equivalent [3], [9], [10]. Both, parallel plane switches and Birkhoff-von-Neumann switches with load balancing are a special case of Clos packet switches with load balancing, where the number of input ports of the input SEs is $n = 1$. The PPS architecture comprises input/output line cards and output-queued switching elements (SEs) in the middle stage. Each line card is connected to each SE in the middle stage, and packets from each line card are balanced over the output queued switches in the middle stage [9]. In PPS, $n = 1$ and $N = m$, where N is the number of the switch ports and m is the number of the SE ports. In other words, the number of the switch ports equals the number of the SE ports. Usually, the SE is implemented on a single memory chip. Therefore, the number of switch ports is limited by the memory pin count. On the other side, the number of switch ports in Clos packet switch is n times larger, and typically $n >> 1$. The number of ports determines the number of switches in the network that a given switch can simultaneously reach. If the switch can reach the smaller number other switches, packets have to pass the larger number of switches and the switching capacity wasted for packet transit increases. Birkhoff-von-Neumann switch comprises two stages of cross-bar fabrics with input buffers. Each input-output pair of the fabric in either stage is allocated equal capacity through it. In this way, the traffic is balanced through the first stage fabric, so that it is uniform when passing the second stage fabric [3]. In this design, a centralized scheduler is omitted, but the high-capacity cross-bars are still required. The total switch capacity is limited by the cross-bar capacity. Also, synchronization on a cell-by-cell basis is needed across the fabric. Ways to simplify the design of high-capacity cross-bars were proposed in [10]. This paper made it clear that PPS and Birkhoff-von-Neumann switch are equivalent architectures. The architecture in [10] again assumes the small number of high-capacity line cards. Such line cards would require involved development. Also, it is not only the capacity that matters, but also the number of switch ports should be large as we noted. However, assumed line-cards can be viewed as shared buffers to which multiple regular capacity line-cards (10Gbps) are attached. A general formula derived in our paper will be applied to assess the performance of the architecture proposed in [10].

Delay sensitive traffic is a significant part of the Internet traffic which is ever increasing. The delay incurred by Clos packet switches based on load balancing has not been previously assessed. In this paper, we examine the fabric utilization under which a tolerable delay can be guaranteed to the most sensitive applications in Clos packet switches based on load balancing. It will be shown that the tolerable delay is guaranteed only for the fabric utilization that unacceptably decreases with the increasing number of flows that are balanced. Parts of this analysis have been presented in [20], [21], [22], [23], [24], [25]. It will turn out that if the end-to-end sessions are balanced independently as proposed in [27], the tolerable delay is guaranteed only for very low utilization. For this reason, we propose novel load balancing algorithms that will be proven to provide a superior performance. First, we will describe several options for load balancing of flows with different granularities: either inputs or input SEs may balance traffic, and flows to either output SEs or outputs may be balanced independently. Formula for the fabric utilization in terms of the number of flows, and tolerable delay will be then derived. Formula for the fabric speedup required to provide the 100$\%$ switch utilization will be also derived in terms of the number of flows and tolerable delay. In addition, we analyze the switch performance when load balancing of different flows is desynchronized. This analysis shows that a significant improvement of the performance can be achieved with the minor increase of the implementation complexity. Based on the presented performance analysis, the adequate switch parameters and load balancing algorithms will be recommended at the end.

II. Terminology

SE$_{ij}$ - switching element j in stage i.

a_{ij} - the number of cells per frame that input i can be guaranteed to transmit to output j.

c_{ij} - the counter which is associated to flow (i, j).

D - a given tolerable delay.

F - the number of cells per frame on the external links.

F'_c - the number of cells per frame passing a link from an input SE to a center SE.

F''_c - the number of cells per frame passing a link from a center SE to an output SE.

F_u - the number of cells per frame that an input is guaranteed to transmit, and an output is guaranteed to receive.

l - the number of center SEs.

m - the number of input and output SEs.

n - the number of inputs of an input SE, and the number of outputs of an output SE.

N - the number of switch inputs and outputs.

N_f - the maximum number of flows that are balanced through some internal fabric link.

N'_f - the number of flows that are balanced through a link from an input SE to a center SE.

N''_f - the number of flows that are balanced through a link from a center SE to an output SE.

R - the bit-rate that an external (input/output) link can support.

R_c - the bit-rate that an internal fabric link can support (a link between either input and center SE, or center and output SE).

S - speedup equal to the ratio of the total capacities of internal and external links.

S_i - the minimum speedup.

S_{ik} - the minimum speedup for the load balancing algorithm k.

S_d - the minimum speedup when the counters are desynchronized.

S_{dk} - the minimum speedup for the load balancing algorithm k of the links when the counters are desynchronized.

T_c - the cell duration, or the time slot duration.

U_i - the maximum utilization.

U'_i - the maximum utilization of the links from input to center SEs.

U''_i - the maximum utilization of the links from center to output SEs.

U_{ik} - the maximum utilization for the load balancing algorithm k.

U_d - the maximum utilization when the counters are desynchronized.

U'_d - the maximum utilization of the links from input to center SEs when the counters are desynchronized.

U''_d - the maximum utilization of the links from center to output SEs when the counters are desynchronized.

U_{dk} - the maximum utilization for the load balancing algorithm k when the counters are desynchronized.

III. Description of Load Balancing Algorithms in the Clos Packet-Switches

Obviously, when cells reach the center SEs (SEs in the second stage), they are further routed according to their output addresses. So, load balancing can be only performed at the input SEs (SEs in the first stage). We will discuss four different load balancing algorithms. They differ according to the definition of the flows that are balanced. For example, in one definition a flow comprises cells sourced by some input and bound to the same output; in another definition a flow comprises cells sourced by some input and bound to the same output SE (SE in the third stage); or a flow comprises cells sourced by the same input SE and bound to the same output; or a flow comprises cells sourced by the same input SE and bound to the same output SE etc.

In the first load balancing algorithm, cells from some input bound for the particular output are spread equally among center SEs. In the second case, cells from some input bound for the particular output SE are spread equally among center SEs. The previous two load balancing algorithms can be implemented when SEs are cross-bars and inputs operate independently. Then, the load can be balanced by input SEs: an arbiter associated with each input SE determines to which center SE a cell will be transmitted. So, in the third algorithm, cells transmitted from an input SE to some output would be spread equally across the center SEs. In the fourth algorithm, cells transmitted from an input SE to some output SE would be spread equally across the center SEs. In the last two algorithms, SEs should be output-queued shared buffers, because multiple incoming packets might have to be assigned to the same queue.

In the first load balancing algorithm, input $i,\ 0 \leq i < N$, has N different counters associated with different outputs, $c_{ij},\ 0 \leq j < N$. Here $N = nm$ is the number of switch input and output ports. A cell arriving to input i and bound for the jth output will be marked to be transmitted through the c_{ij}th output of its SE, i.e. to be transmitted through the c_{ij}th center SE. Then, the counter in question is incremented modulo l, namely $c_{ij} \leftarrow (c_{ij} + 1) \bmod l$. In the second load balancing algorithm, input $i,\ 0 \leq i < N$, stores m counters associated with different switch output SEs, $c_{ij},\ 0 \leq j < m$. In the third load balancing algorithm, input SE $i,\ 0 \leq i < m$, stores N different counters associated with different outputs, $c_{ij},\ 0 \leq j < N$. In the fourth load balancing algorithm, input SE $i,\ 0 \leq i < m$, stores m counters associated with different switch output SEs, $c_{ij},\ 0 \leq j < m$. In all cases, a cell of some flow will be marked to be transmitted through the c_{ij}th output of its SE, i.e. to be transmitted through the c_{ij}th center SE, where c_{ij} is the counter corresponding to the flow in question. The counter value is then incremented modulo l.

The SE based on a cross-bar is shown in Figure 2 (a). This is the input SE considering its number of input and output ports. A header reader and routing decision (HR&RD) block reads a packet header, and stores the packet in the appropriate virtual output queue (VOQ) of an input buffer. The packet is then scheduled and transmitted from the SE. In the first stage, the HR&RD block reads the packet address, determines to which flow the packet belongs to, and reads the counter of that flow from the counter lookup table. The counter value determines the center SE through which the packet will be routed, i.e. the VOQ of the input buffer where the packet should be stored. In the second stage, the HR&RD block determines the output SE to which the packet should be sent solely based on the packet header. The SE based on a shared buffer is shown in Figure 2 (b). It is identical to the SE based on a cross bar only the input buffers and the cross-bar are replaced by the shared buffer.

Let us examine the blocking nature of a Clos packet switch based on the load balancing. As we noted before, the algorithms for SE configuration in Clos circuit switches are not applicable to Clos packet switches where configurations are changed fast, on a cell-by-cell basis. So, the proofs for non-blocking conditions in Clos circuit switches do not hold for the architecture in question. In packet switches the internal links of the fabric often have higher capacity than the external links, in order to provide non-blocking. A fabric speedup can be defined as the ratio of the capacities of the internal links and the external links:

$$S = \frac{nmR_c}{mlR} \geq 1, \tag{1}$$

where R is the maximum bit-rate supported by a link that is attached to the switch port, and R_c is the maximum bit-rate

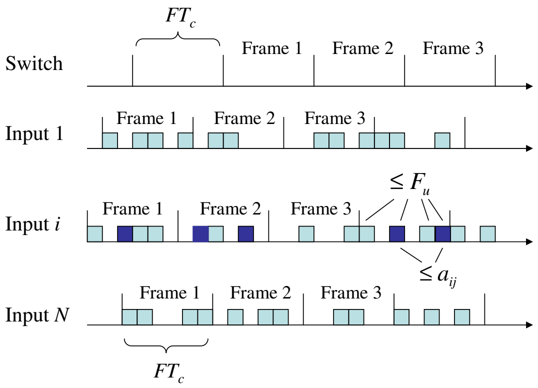


Fig. 3. Time diagram explaining a coarse synchronization.

supported by a fabric internal link that either connects input and center SE, or center and output SE.

Theorem 1: Non-blocking is provided in Clos packet switches based on load balancing without the fabric speedup, i.e. the traffic passes the fabric as long as outputs are not overloaded and $S \geq 1$.

Proof: It is easy to see that the traffic loads passing through the internal links of the same SE are identical when the described load balancing algorithms are applied. Because the total traffic passing through any SE does not exceed but can reach nR, the traffic passing some internal link does not exceed but can reach nR/l. This traffic will pass the link if and only if $nR/l \leq R_c$, which is true if and only if $S \geq 1$. ■

IV. Generalized Performance Analysis of Load Balancing Algorithms

Traffic of each individual flow is balanced independently across the SEs. If there are many flows that transmit cells across some SE at the same time, the cells will experience long delay. Many applications, e.g. voice and video, require rate and delay guarantees. We will assess the worst case utilizations for balancing algorithms that provide rate and delay guarantees. We will focus on the traffic that requires rate and delay guarantees, and assume that this traffic is policed at either the edge of the network, or at the switch ports. Note that policing is necessary for rate and delay guarantees to be provided. For example, input 1 negotiated to send 10Mbps to output 3, the policing interval is 0.5ms, the cell duration is 50ns and the port bit rate is 10Gbps. Then, input 1 will send at most 10 high-priority cells per frame to output 3. We also assume that SEs are non-blocking and provide rate and delay guarantees. So, they transfer all the policed traffic within one frame period. These features hold when the shared buffers are used as SEs. But they also hold for the cross-bar SEs with the speedup of two that are run by the maximal matching algorithms [15], [16], [19], [26].

Time of a switch and its input ports is divided into the policing intervals, or frames, that are FT_c long as shown in Figure 3, where F is the number of time slots of duration T_c. Each input-output pair is guaranteed a specified number of time slots per frame. For example a_{ij} time slots are guaranteed to input-output pair (i,j), $0 \leq i,j < N$. Each input, and each output can be assigned at most F_u time slots per frame, i.e.

$$\sum_k a_{ik} \leq F_u, \;\; \sum_k a_{ki} \leq F_u. \tag{2}$$

We will evaluate F_u in terms of F, N, N_f for various load balancing algorithms, so that all cells of a frame pass each switch stage within one frame. Here N_f is the maximum number of flows that are balanced through some connection, and N is the number of switch ports.

We assume that there is a coarse synchronization in the switch, i.e. that all SEs use identical frame delineation. The synchronization is coarse because it is performed on a frame-by-frame basis and not on a cell-by-cell basis. The coarse synchronization is shown in Figure 3. The first time axis shows switch frames in which SEs forwards cells of the corresponding input frames shown on the axes below. Frame delineations for different input ports may vary, and the input frames with the same ordinal number overlap while preceding the switch frame with the same ordinal number. In a particular switch frame of the first time axis in Figure 3, the input SEs will pass cells of the input frames with the same ordinal number, the center SEs will pass cells of the input frames with the ordinal number decremented by 1, and the output SEs will pass cells of the input frames with the ordinal number decremented by 2. For example, in the switch frame 3 the input SEs pass the cells that have arrived in the input frames designated by 3, the center SEs pass the cells that have arrived in the input frames 2, and the output SEs pass the cells that have arrived in the input frames 1. This coarse synchronization can simplify the controller implementation. Otherwise, SEs should give priority to the earlier frames which complicates their schedulers, also cell resequencing becomes more complex because the maximum jitter is increased. We will calculate the fabric utilization such that all cells of a frame are guaranteed to pass the switch stage within the next frame, so resequenced cells may be at most F cells apart. Consequently the resequencing buffer size is F cells, and the total cell delay is increased for FT_c. The total delay that a cell may experience through a three-stage Clos packet switch is four times the frame duration:

$$D = 4FT_c. \tag{3}$$

We will calculate the number of cells passing through an internal fabric link per frame in terms of F_u, F (or $D/4T_c$), N, N_f which should be smaller than the number of cells per frame on the internal link equal to SFn/l, and from this inequality we will calculate the maximum utilization of the input links (F_u/F). Also, we will calculate the speedup needed to achieve the 100% switch utilization, and a given tolerable delay. Note that all lemmas and theorems hold in large switches where $l > 10$.

A. Switch Utilization

Lemma 1: Let F_c' denote the maximum number of cells per frame sent from a given input SE through a given center SE. It holds that

$$\frac{nF_u}{l} + N_f' - n \leq F_c' < \frac{nF_u}{l} + N_f', \tag{4}$$

where N'_f denotes the maximum number of flows sourced by input SE that pass through the links from this SE to center SEs.

Proof: Let f'_{ig}, $0 \le g < N'_f$, denote the number of time slots per frame that are guaranteed to the individual flows sourced by SE_{1i}. It follows:

$$\begin{aligned} F'_c &\le \sum_g \left\lceil \frac{f'_{ig}}{l} \right\rceil \Rightarrow \\ F'_c &< \sum_g \frac{f'_{ig}}{l} + N'_f \Rightarrow \\ F'_c &< \frac{nF_u}{l} + N'_f, \end{aligned} \tag{5}$$

where $\lceil x \rceil$ is the smallest integer no less than x, i.e. $\lceil x \rceil < x+1$. This proves the right side of inequality (4). We can find a case in which the number of cells passing an internal link per frame exceeds the left side of inequality (4), and so does F'_c according to its definition. Assume that out of N'_f flows sourced by SE_{1i}, $N'_f - n$ flows are assigned one time slot per frame, and the remaining n flows are assigned $nF_u - (N'_f - n)$ time slots per frame. If it happens that first cells in a frame of all flows are sent through SE_{2j}, the total number of cells per frame transmitted through SE_{2j} from SE_{1i} will be:

$$\begin{aligned} F'_c &= N'_f - n + n\lceil \frac{F_u}{l} - \frac{N'_f}{nl} \rceil \\ &= \frac{nF_u}{l} + \frac{(l-1)N'_f - (nF_u - N'_f) \bmod (nl)}{l} \Rightarrow \\ F'_c &\ge \frac{nF_u}{l} + N'_f - n \end{aligned} \tag{6}$$

for $l > 10$. Claim of the lemma follows. ■

Lemma 2: Maximum utilization of the links from input to center SEs, U'_i, satisfies inequality:

$$\max(0, S - \frac{lN'_f}{nF}) < U'_i \le \min(1, S - \frac{lN'_f}{nF} + \frac{l}{F}). \tag{7}$$

Proof: Note that nSF/l is the number of cells that may pass the link from an input to a center SE within one frame. Let F_{uc} is such that:

$$\frac{nF_{uc}}{l} + N'_f = \frac{nSF}{l}. \tag{8}$$

If F_{uc} is the number of cells that are guaranteed to an input or to an output per frame, the number of cells passing an internal link satisfies

$$F'_c \le \frac{nF_{uc}}{l} + N'_f = nSF/l, \tag{9}$$

according to lemma 1, and all the cells will pass the internal links in question within a frame. So the maximum utilization under which all cells pass the switch is $U'_i \ge F_{uc}/F$ and the left side of inequality in Lemma 2 is proven. From Lemma 1 $F'_c \ge nF_u/l + N'_f - n$, so it must hold

$$\begin{aligned} \frac{nF_u}{l} + N'_f - n &\le F'_c \le \frac{nSF}{l} \Rightarrow \\ U'_i \le \frac{F_u}{F} &\le S - \frac{lN'_f}{nF} + \frac{l}{F}, \end{aligned} \tag{10}$$

and the right side of inequality in Lemma 2 is proven. ■

Lemma 3: Let F''_c denote the maximum number of cells per frame sent to a given output SE through a given center SE. It holds that

$$\frac{nF_u}{l} + N''_f - n \le F''_c < \frac{nF_u}{l} + N''_f, \tag{11}$$

where N''_f denotes the maximum number of flows bound to some output SE that pass through the links from center SEs to this output SE.

Proof: The proof is similar to the proof of Lemma 1. ■

Lemma 4: Maximum utilization of the links from center to output SEs is:

$$\max(0, S - \frac{lN''_f}{nF}) < U''_i \le \min(1, S - \frac{lN''_f}{nF} + \frac{l}{F}). \tag{12}$$

Proof: The proof is similar to the proof of Lemma 2. ■

Theorem 2: Maximum utilization of the fabric internal links under which all cells pass them within designated frames is:

$$\max(0, S - \frac{lN_f}{nF}) \le U_i \le \min(1, S - \frac{l(N_f - n)}{nF}). \tag{13}$$

where N_f is the maximum number of flows that are passing through some internal link of the fabric.

Proof: The proof follows from Lemmas 2 and 4. ■

B. Switch Utilization when the Counters are Desynchronized

We calculated the maximum utilization when different flows bound for the same SE are independently balanced, so the cells of a given frame are sent starting from the same center SE. Alternatively, equal numbers of flows are balanced starting from different center SEs in each frame. For example, flow g of SE_{1i} resets its counter at the beginning of a frame to $c_{ig} = (i+g) \bmod l$. Or, flow g bound to SE_{3k} resets its counter at the beginning of a frame to $c_{kg} = (k+g) \bmod l$. We can assume $N'_f, N''_f > 10l$ or $N'_f = N''_f = 0 \bmod l$ in order to simplify the analysis of load balancing algorithms with the desynchronized counters, due to the fact that other cases will not be of interest in the later discussion.

Lemma 5: In load balancing algorithms with the desynchronized counters, the maximum number of cells passing through a link from an input SE to a center SE is:

$$F'_c = \begin{cases} \frac{nF_u}{l} + \frac{N'_f}{2} & F \ge \frac{lN'_f}{2n} \\ \sqrt{\frac{2nF_uN'_f}{l}} & F < \frac{lN'_f}{2n}. \end{cases} \tag{14}$$

Proof: We will calculate the maximum number of cells that are transmitted from SE_{1i} through $\mathrm{SE}_{2(n-1)}$ in the middle stage, and the same result would hold for any other center SE. Let f'_{ig} denote the number of cells in flow g which is balanced starting from SE_{2j} at the beginning of each frame, where $j = (i+g) \bmod l$. Then, the number of cells in flow g transmitted from SE_{1i} through $\mathrm{SE}_{2(n-1)}$ is $\left\lfloor (f'_{ig} + (i+g) \bmod l)/l \right\rfloor$, where

$\lfloor x \rfloor$ is the smallest integer not greater than x i.e. $\lfloor x\rfloor \le x$. So, the number of cells from SE_{1i} through SE$_{2(n-1)}$ is:

$$\begin{aligned} F'_c &= \sum_{0\le g<N'_f} \left\lfloor \frac{f'_{ig}+(i+g) \bmod l}{l}\right\rfloor \\ &\le \sum_{0\le g<N'_f} \frac{f'_{ig}+(i+g)\bmod l}{l} \\ &\approx \frac{nF_u}{l}+\frac{N'_f}{l}\cdot\frac{l-1}{2} \\ &\approx \frac{nF_u}{l}+\frac{N'_f}{2}, \end{aligned} \tag{15}$$

for $l>10$ and $N'_f>10l$. Note that inequality (15) holds for $l>10$ and $N'_f \bmod l=0$ as well. Equality in (15) is reached if and only if:

$$f'_{ig}=l-(i+g) \bmod l + l\cdot y'_{ig}, \tag{16}$$

where $y'_{ig}\ge 0$ are integers. Values f'_{ig} that satisfy condition (16) exist if it holds that:

$$\begin{aligned} nF_u &= \sum_{0\le g<N'_f} f'_{ig} \\ &\ge \sum_{0\le g<N'_f} (l-(i+g) \bmod l)=\frac{N'_f}{l}\cdot\frac{l(l+1)}{2}\Leftrightarrow \\ F_u &\ge \frac{N'_f}{n}\cdot\frac{l+1}{2}\approx\frac{lN'_f}{2n}, \end{aligned} \tag{17}$$

for $l>10$ and $N_f>10l$. Note that inequality (17) holds for $l>10$ and $N_f \bmod l=0$ as well. When inequality (17) holds, equality in (15) may be reached, and:

$$F'_c=\frac{nF_u}{l}+\frac{N'_f}{2}. \tag{18}$$

If inequality (17) does not hold:

$$\begin{aligned} \frac{N'_f}{l}\cdot\frac{z(z+1)}{2} &\le nF_u<\frac{N'_f}{l}\cdot\frac{(z+1)\cdot(z+2)}{2}\Leftrightarrow \\ z &= \left\lfloor\frac{-1+\sqrt{1+\frac{8nlF_u}{N'_f}}}{2}\right\rfloor, \end{aligned} \tag{19}$$

where $0\le z<l$ is an integer. Because $F_u\ge 10N'_f/(8nl)$:

$$z\approx\sqrt{\frac{2nlF_u}{N'_f}}. \tag{20}$$

It is easy to understand that F'_c will be maximal for:

$$f'_{ig}=\begin{cases} l-q & l-z\le q=(i+g) \bmod l<l \\ 0 & 0\le (i+g) \bmod l<l-z. \end{cases} \tag{21}$$

If $F_u<lN'_f/(2n)$, from (15,20,21):

$$F'_c=\frac{N'_f z}{l}\approx\sqrt{\frac{2nF_uN'_f}{l}}. \tag{22}$$

■

Lemma 6: Maximum utilization of the links from input to center SEs, when the counters are desynchronized is:

$$U'_d=\begin{cases} S-\frac{lN'_f}{2nF} & F\ge\frac{lN'_f}{nS} \\ \frac{nS^2F}{2lN'_f} & F<\frac{lN'_f}{nS}. \end{cases} \tag{23}$$

Proof: Since $F'_c\le nSF/l$, from Lemma 5 it follows that for $F_u\ge lN'_f/(2n)$,

$$\begin{aligned} F'_c &= \frac{nF_u}{l}+\frac{N'_f}{2}\le\frac{nSF}{l}\Rightarrow \\ U'_d &= \frac{F_u}{F}\le S-\frac{lN'_f}{2nF} \\ F &\ge \frac{lN'_f}{nS}, \end{aligned} \tag{24}$$

and for $F_u<lN'_f/(2n)$:

$$\begin{aligned} F'_c &= \sqrt{\frac{2nF_uN'_f}{l}}\le\frac{nSF}{l}\Rightarrow \\ U'_d &= \frac{F_u}{F}\le\min(\frac{lN'_f}{2nF},\frac{nS^2F}{2lN'_f}). \end{aligned} \tag{25}$$

So, the maximum utilization when counters are reset each frame is:

$$\begin{aligned} U'_d &= \frac{F_u}{F} \\ &\le \begin{cases} S-\frac{lN'_f}{2nF} & F_u\ge\frac{lN'_f}{2n} \\ \min(\frac{lN'_f}{2nF},\frac{nS^2F}{2lN'_f}) & F_u<\frac{lN'_f}{2n}. \end{cases} \end{aligned} \tag{26}$$

From equations (24,26), it follows that:

$$U'_d=\begin{cases} S-\frac{lN'_f}{2nF} & F\ge\frac{lN'_f}{nS} \\ \frac{nS^2F}{2lN'_f} & F<\frac{lN'_f}{nS}. \end{cases} \tag{27}$$

■

Lemma 7: In load balancing algorithms with the desynchronized counters, the maximum number of cells passing a link from a center SE to an output SE is:

$$F''_c=\begin{cases} \frac{nF_u}{l}+\frac{N''_f}{2} & F\ge\frac{lN''_f}{2n} \\ \sqrt{\frac{2nF_uN''_f}{l}} & F<\frac{lN''_f}{2n}. \end{cases} \tag{28}$$

Proof: Let f''_{kg} denote the number of cells in flow g transmitted to SE$_{3k}$ that are balanced starting from SE$_{2j}$ at the beginning of each frame, where $j=(k+g) \bmod l$. Then, the number of cells in flow g transmitted to SE$_{3k}$ through SE$_{2(n-1)}$ is $\lfloor(f''_{kg}+(k+g) \bmod l)/l\rfloor$. The rest of the proof is similar to the proof of Lemma 5. ■

Lemma 8: Maximum utilization of the links from center to output SEs when the counters are reset each frame is:

$$U''_d=\begin{cases} S-\frac{lN''_f}{2nF} & F\ge\frac{lN''_f}{nS} \\ \frac{nS^2F}{2lN''_f} & F<\frac{lN''_f}{nS}. \end{cases} \tag{29}$$

Proof: The proof is similar to the proof of Lemma 6. ∎

Theorem 3: In the algorithms where balancing of different flows is desynchronized, maximum utilization of the fabric internal links under which all cells pass it within designated frames is:

$$U_d = \begin{cases} S - \frac{lN_f}{2nF} & F \geq \frac{lN_f}{nS} \\ \frac{nS^2F}{2lN_f} & F < \frac{lN_f}{nS}, \end{cases} \tag{30}$$

where N_f is the maximum number of flows balanced through some fabric internal link.

Proof: Maximum utilization of the fabric internal links under which all cells pass it within designated frames is derived from Lemmas 6 and 8 to be:

$$U_d = \min(U_d', U_d'') = \begin{cases} S - \frac{lN_f}{2nF} & F \geq \frac{lN_f}{nS} \\ \frac{nS^2F}{2lN_f} & F < \frac{lN_f}{nS}. \end{cases} \tag{31}$$

∎

Note that Theorem 3 provides the maximum utilization when both balancing of flows sourced by an input SE, and balancing of flows bound for an output SE are desynchronized. This assumption will hold in all consider algorithms.

C. Switch Speedup with and without Desynchronized Counters

Often, signal transmission over the fibers connecting distant routers requires most complex and costly hardware. Therefore, it is important to provide the highest utilization of the fiber transmission capacity. For this reason, switching fabrics with the speedup have been previously proposed and used.

Theorem 4: The speedup S required to pass all incoming packets with a tolerable delay when the counters are changing independently is:

$$1 + \frac{l(N_f - n)}{nF} \leq S_i < 1 + \frac{lN_f}{nF}, \tag{32}$$

and the speedup when counters are desynchronized is:

$$S_d \geq \begin{cases} 1 + \frac{lN_f}{2nF} & F \geq \frac{lN_f}{2n} \\ \sqrt{\frac{2lN_f}{nF}} & F < \frac{lN_f}{2n}. \end{cases} \tag{33}$$

Proof: It should hold that $F_u = F$ while $F_c \leq nSF/l$, where F_c is the number of cells passing through some internal link per frame. When the counters are independent from Lemmas 1 and 3 it follows that:

$$\frac{nS_iF}{l} \geq \max\left(F_c', F_c''\right) \geq \frac{nF}{l} + N_f - n, \tag{34}$$

which proves the left hand side of inequality (32). If speedup S_{ic} is such that:

$$\frac{nS_{ic}F}{l} = \frac{nF}{l} + N_f,$$

then, from Lemmas 1 and 3 it follows that:

$$\frac{nS_{ic}F}{l} = \frac{nF}{l} + N_f \geq \max(F_c', F_c''),$$

and all the traffic is guaranteed to pass the fabric with the speedup S_{ic}, which proves the right hand side of inequality (32).

When the counters are desynchronized, from Lemmas 5 and 7 it follows that:

$$\frac{nS_dF}{l} \geq \max\left(F_c', F_c''\right) = \begin{cases} \frac{nF}{l} + \frac{N_f}{2} & F \geq \frac{lN_f}{2n} \\ \sqrt{\frac{2nFN_f}{l}} & F < \frac{lN_f}{2n}, \end{cases}$$

and so inequality (33) follows. ∎

V. Performance of Load Balancing Algorithms

In this section, the switch performance for various system parameters will be discussed. All graphs are drawn according to the previously derived formulas. These formulas accurately describe the worst case switch performance. So, the switch performance for any particular traffic pattern will not be worse than the performance shown in the graphs. Since the worst case performance in the graphs can be reached for certain traffic patterns, it determines which load balancing schemes are acceptable and which are not.

We will discuss the switch utilization and the fabric speedup for a given tolerable delay. One way packet delay that can be tolerated by interactive applications is around 150ms, but only 50-60ms of this allowed delay can be budgeted for the queueing. The switch delay below 3ms may be required for various reasons. For example, packets might pass multiple packet switches from their sources to the destinations, and packet delays through these switches would add. Also, in order to provide flexible multicasting, the ports should forward packets multiple times through the packet switch, and the packet delay is prolonged accordingly [2], [17], [18], [27].

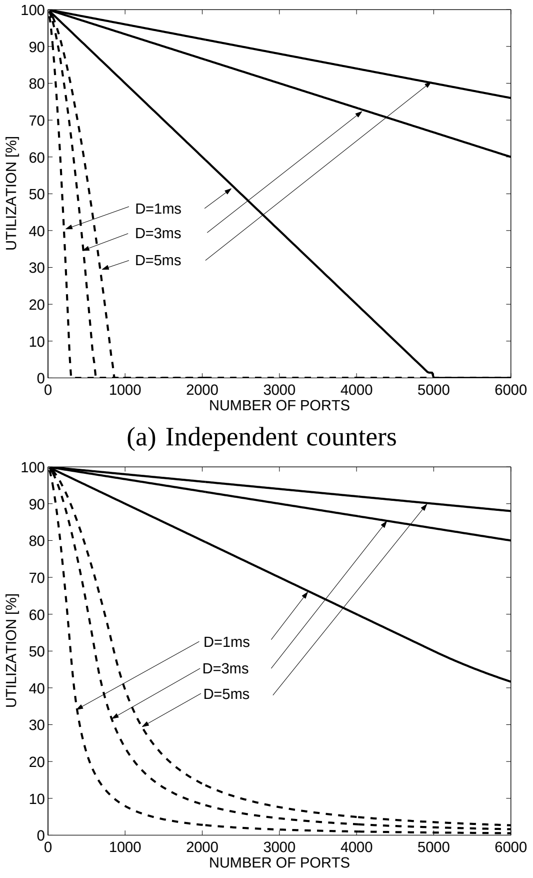
It can be observed from our previous analysis that the performance of a load balancing algorithm depends on the number of balanced flows. Let N_f denote the maximum number of balanced flows passing through some internal link. N_f is equal to the maximum number of flows sourced by some input SE or bound to some output SE.

First we will assume that the Clos packet switch comprises identical $n \times n$ SEs, i.e. that $n = m = l = \sqrt{N}$. In the first algorithm, $N_f = nN$, because any input SE sources nN flows, and each of N inputs balances n flows bound for any output SE. In the second algorithm $N_f = N$, because any input SE sources $n^2 = N$ flows, and each of N inputs balances one flow for any output SE. In the third algorithm, $N_f = N$ because any input SE sources N flows, and each of n input SEs balances n flows for any output SE. In the fourth algorithm, $N_f = n$ because any input SE sources n flows, and each of n input SEs balances one flow for any output SE.

Under the assumption of no speedup, i.e. $S = 1$, we obtain the maximum utilizations for described load balancing algorithms by substituting N_f in formula (13):

$$\begin{aligned} U_{i1} &= \max(0, 1 - \frac{nN}{F}), \\ U_{i2} &= U_{i3} = \max(0, 1 - \frac{N}{F}), \\ U_{i4} &\approx 1. \end{aligned} \tag{35}$$

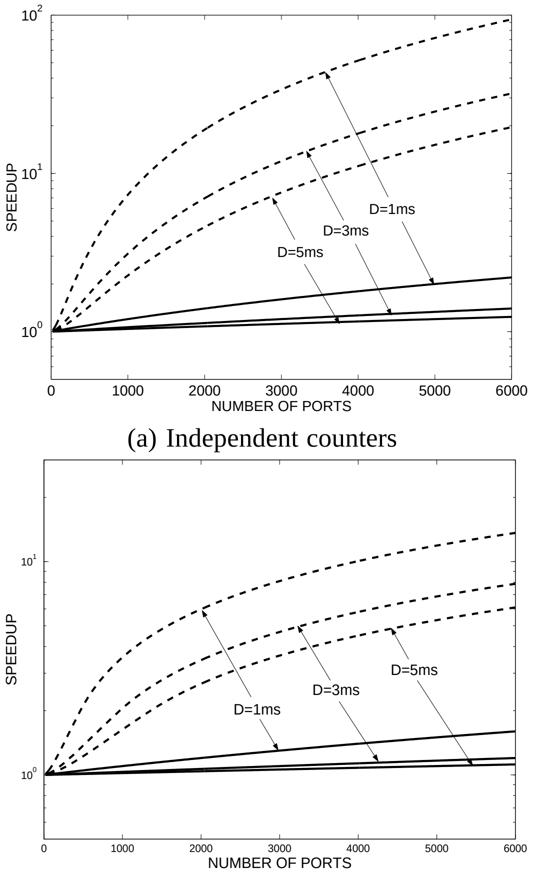
So, the first load balancing algorithm is least efficient, while the fourth algorithm is most efficient. However, the fourth load balancing algorithm is not an obvious design choice because



(a) Independent counters

(b) Desynchronized counters

Fig. 4. Switch utilization when counters are (a) independent, (b) desynchronized: solid curves represent the algorithm in which inputs balance flows bound for output SEs, and to the algorithm in which input SEs balance flows bound for outputs; dashed curves correspond to the algorithm in which inputs balance flows bound for outputs.



(a) Independent counters

(b) Desynchronized counters

Fig. 5. Fabric speedup when the counters are (a) independent, (b) desynchronized: solid curves represent the algorithm in which inputs balance flows bound for output SEs, and to the algorithm in which input SEs balance flows bound for outputs; dashed curves correspond to the algorithm in which inputs balance flows bound for outputs.

it requires the shared buffers, while the first two algorithms require the cross-bars that are more scalable.

In order to increase the efficiency of the load balancing algorithms, the frame length should be increased. On the other side, the cell delay is proportional to the frame length. So the maximum frame length will be determined by the delay that could be tolerated by the applications such as interactive voice and video. Assume that the maximum delay that can be tolerated by interactive applications is D, and the cell time slot duration is T_c, then

$$F = \frac{D}{4T_c} \tag{36}$$

and:

$$\begin{aligned} U_{i1} &= \max(0, 1 - \frac{4nNT_c}{D}), \\ U_{i2} = U_{i3} &= \max(0, 1 - \frac{4NT_c}{D}). \end{aligned} \tag{37}$$

If flows are balanced starting from different center SEs, the efficiency of load balancing can be improved. Namely, at the beginning of each frame, counters will be set to the appropriate values, e.g. $c_{ij} = (i + j) \bmod l$, where $0 \le i, j < N$ for the first load balancing algorithm, $0 \le i < N,\ 0 \le j < n$ for the second algorithm, $0 \le i < n,\ 0 \le j < N$ for the third algorithm. (Efficiency of the third algorithm is already close to 100%.) Because in all these cases $N_f \ge 10n$ and $n > 10$, the guaranteed utilizations for the enhanced load balancing algorithms can be derived by substituting N_f in formula (30):

$$U_{d1} = \begin{cases} 1 - \frac{nN}{2F} & F \ge nN \\ \frac{F}{2nN} & F < nN, \end{cases}$$

$$U_{d2} = U_{d3} = \begin{cases} 1 - \frac{N}{2F} & F \ge N \\ \frac{F}{2N} & F < N. \end{cases} \tag{38}$$

$$\tag{39}$$

It follows that:

$$U_{d1} = \begin{cases} 1 - \frac{2nNT_c}{D} & D \ge 4nNT_c \\ \frac{D}{8nNT_c} & D < 4nNT_c, \end{cases}$$

$$U_{d2} = U_{d3} = \begin{cases} 1 - \frac{2NT_c}{D} & D \ge 4NT_c \\ \frac{D}{8NT_c} & D < 4NT_c. \end{cases} \tag{40}$$

where D is the maximum delay that can be tolerated, and again it is assumed that there is no speedup, i.e. that $S = 1$.

Figure 4 shows the fabric utilization decrease as the switch size is increasing for various tolerable delays. In 4 (a) counters are independent, while in (b) counters are desynchronized. The cell duration is 50ns. The solid curves represent the second and the third algorithm ($N_f = N$), while the dashed curves correspond to the first algorithm ($N_f = nN$). One can see that the efficiency of the first balancing algorithm might decrease unacceptably as the switch size is increasing. For example, the utilization of a fabric with 1000 ports drops below 10$\%$ for a tolerable delay of 3ms. On the other side, for the same tolerable delay and cell duration, the utilization of a fabric with 4000 ports is 80$\%$ if the second or the third load balancing algorithm is applied. It can be concluded that the last three load balancing algorithms (for which $N_f \leq N$) provide a superior performance. We note that the efficiency of the first load balancing algorithm is improved when the counters are desynchronized, but, it is still low in the large switches where cells bound for the particular output are spread equally across the center SEs. For example, the utilization of a fabric with 1000 ports drops below 30$\%$ for a tolerable delay of 3ms, and below 10$\%$ in a switch with 4000 ports. The efficiency of the second and the third load balancing algorithm is improved too, for the same tolerable delay, the utilization of a fabric with 4000 ports is 90$\%$.

If the utilization of the transmission capacity is to be maximized to 100$\%$, the switching fabric with a speedup should be implemented. The speedup required to provide the 100$\%$ utilization varies for different load balancing algorithms. In the simple case when different counters are independent, required speedups can be obtained from formula (32) to be:

$$
\begin{aligned}
S_{i1} &= 1+\frac{nN}{F},\\
S_{i2} = S_{i3} &= 1+\frac{N}{F}.
\end{aligned}
\tag{41}
$$

When the counters are desynchronized, the required speedups are decreased and can be obtained from formula (33) to be:

$$
\begin{aligned}
S_{d1} &= \begin{cases} 1+\frac{nN}{2F} & F \geq \frac{nN}{2} \\ \sqrt{\frac{2nN}{F}} & F < \frac{nN}{2}, \end{cases}\\
S_{d2} = S_{d3} &= \begin{cases} 1+\frac{N}{2F} & F \geq \frac{N}{2} \\ \sqrt{\frac{2N}{F}} & F < \frac{N}{2}. \end{cases}
\end{aligned}
\tag{42}
$$

Speedups required to pass the packets with a tolerable delay of D can be calculated from formula (41):

$$
\begin{aligned}
S_{i1} &= 1+\frac{4nNT_c}{D},\\
S_{i2} = S_{i3} &= 1+\frac{4NT_c}{D}.
\end{aligned}
\tag{43}
$$

When the counters are desynchronized, required speedups are decreased and can be obtained from formula (42) to be:

$$
\begin{aligned}
S_{d1} &= \begin{cases} 1+\frac{2nNT_c}{D} & D \geq 2nNT_c \\ \sqrt{\frac{8nNT_c}{D}} & D < 2nNT_c, \end{cases}\\
S_{d2} = S_{d3} &= \begin{cases} 1+\frac{2NT_c}{D} & D \geq 2NT_c \\ \sqrt{\frac{8NT_c}{D}} & D < 2NT_c. \end{cases}
\end{aligned}
\tag{44}
$$

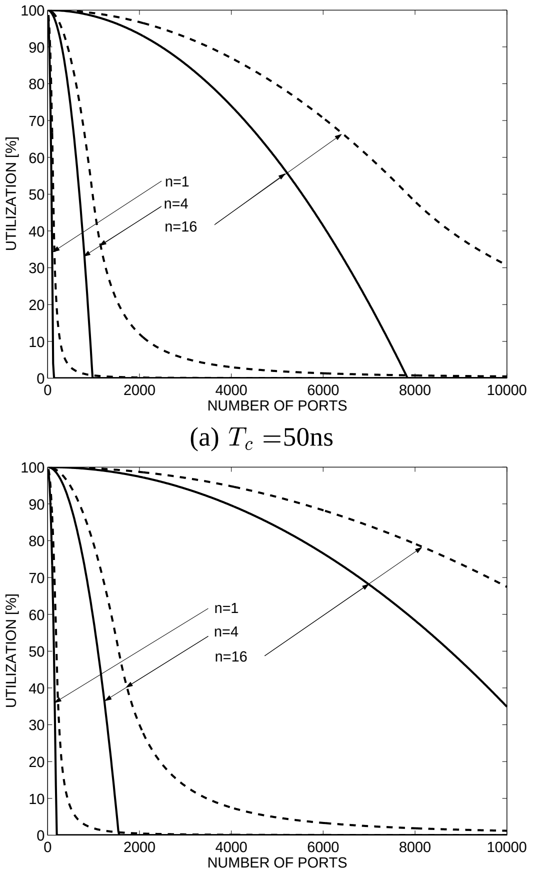
Figure 5 shows the fabric speedup that provides non-blocking through a switch for various delays requirements. In 5 (a) counters are independent, while in (b) counters are desynchronized. The cell duration is 50ns. The solid curves represent the second and the third algorithm ($N_f = N$), while the dashed curves correspond to the first algorithm ($N_f = nN$). The first load balancing algorithm requires the speedups larger than 2 and 10, in order to provide the delay less than 3ms through a switch with 1000 and 4000 ports, respectively. On the other side, the speedup required when the second and third load balancing algorithms are applied is close to 1 for all switch sizes. Figure 5 (b) shows the fabric speedup that provides non-blocking through a switch for various delays requirements in the case when the counters used for balancing are desynchronized. The first load balancing algorithm requires the speedups larger than 2 and 7, in order to provide the delay less than 3ms through a switch with 1000 and 4000 ports, respectively. So, the required speedup is reduced when the counters are desynchronized. No speedup is needed when the second and third load balancing algorithms are applied and the counters are desynchronized.

We observed earlier that the fourth algorithm achieves the best performance. However, this algorithm should be implemented using shared buffers as SEs which are not scalable. Now, we will assume that the SEs are shared buffers of a limited size. The large number of these SEs are required to build high-capacity packet switch, i.e. $l = m > n$. From equations (13,30,32,33) it follows that the performance degrades as l/n increases, but the smaller number of flows that are balanced in the fourth algorithm may compensate for this degradation. As a reminder, in the fourth algorithm a flow comprises cells sourced by the same input SE and bound for the same output SE. If the fourth load balancing is applied, and the switch performance in terms of switch parameters n and N can be calculated using formulas (13,30,32,33) as follows:

$$
\begin{aligned}
U_{i4} &= \max(0, 1-\frac{N^2}{n^3F}),\\
U_{d4} &= \begin{cases} 1-\frac{N^2}{2n^3F} & F \geq \frac{N^2}{n^3} \\ \frac{n^3F}{2N^2} & F < \frac{N^2}{n^3}, \end{cases}\\
S_{i4} &= 1+\frac{N^2}{n^3F},\\
S_{d4} &= \begin{cases} 1+\frac{N^2}{2n^3F} & F \geq \frac{N^2}{2n^3} \\ \sqrt{\frac{2N^2}{n^3F}} & F < \frac{N^2}{2n^3}. \end{cases}
\end{aligned}
\tag{45}
$$

The fabric utilization or the fabric speedup for 100$\%$ utilization for a tolerable delay of D is:

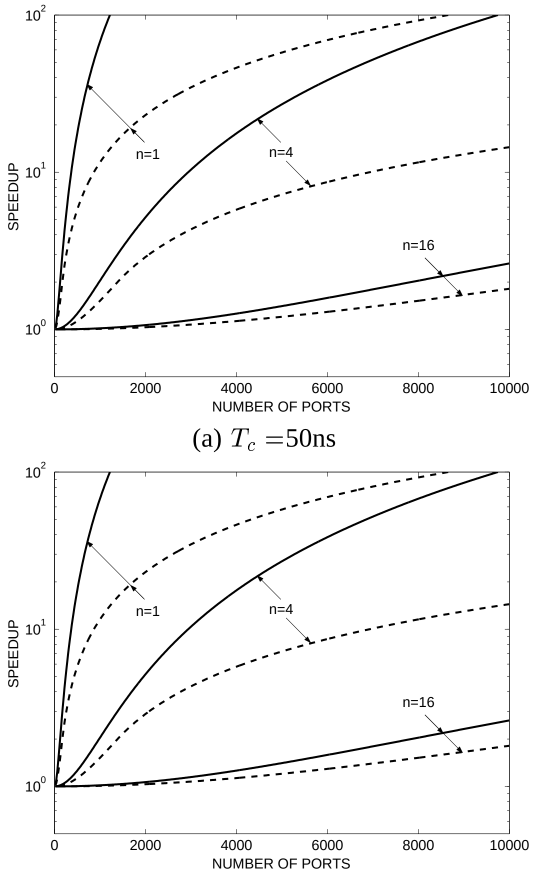
$$
\begin{aligned}
U_{i4} &= \max(0, 1-\frac{4N^2T_c}{n^3D}),\\
U_{d4} &= \begin{cases} 1-\frac{2N^2T_c}{n^3D} & D \geq \frac{4N^2T_c}{n^3} \\ \frac{n^3D}{8N^2T_c} & D < \frac{4N^2T_c}{n^3}, \end{cases}\\
S_{i4} &= 1+\frac{4N^2T_c}{n^3D},
\end{aligned}
$$



(a) T_c =50ns

(b) T_c =20ns

Fig. 6. Switch utilization: solid curves represent the performance with independent counters; dashed curves correspond to the performance with desynchronized counters.



(a) T_c =50ns

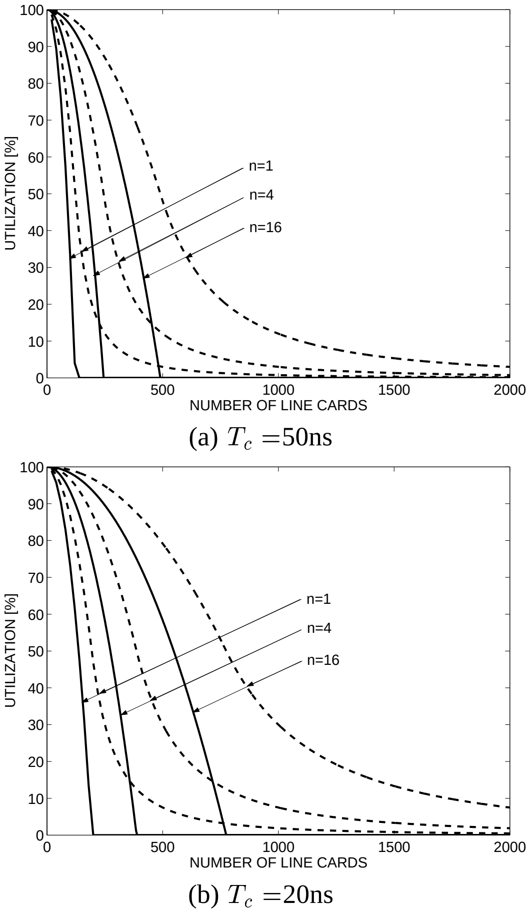
(b) T_c =20ns

Fig. 7. Fabric speedup: solid curves represent the performance with independent counters; dashed curves correspond to the performance with desynchronized counters.

$$S_{d4} = \begin{cases} 1+\frac{2N^2T_c}{n^3D} & D \geq \frac{2N^2T_c}{n^3} \\ \sqrt{\frac{8N^2T_c}{n^3D}} & D < \frac{2N^2T_c}{n^3}. \end{cases} \quad (46)$$

Figure 6 shows the fabric utilization decrease with the number of ports for various numbers of ports per SE. Figure 7 shows the fabric speedup increase with the number of ports for various numbers of ports per SE. The solid curves represent the algorithms in which the counters change independently, while the dashed curves correspond to the algorithms in which the counters are desynchronized. It can be observed that the performance is considerably improved when the counters are desynchronized. For small n, the efficiency drops fast as the number of ports increases. It is non-negligible for the larger switches only when the cell duration is very short and the counters are desynchronized. For example, when N >1000 and $n \in \{1,4\}$, the utilization is close to 0 for $T_c = 50$ns. Utilization improves and exceeds 70$\%$ for $n = 4$ and $T_c = 20$ns. Utilization is above 50$\%$ in the switches with more than 2000 ports only when $n = 16$. Similarly, the speedup required in PPS with $n = 1$ would be high. For N >1000, the required speedup is above 10. On the other side, in regular Clos packet switches where $n \in \{4,16\}$ the fabric utilization and the speedup required for the 100$\%$ switch utilization are improved. The required speedup when $N = 2000$ and $n = 4$ is 5 for $T_c = 50$ns and 4 for $T_c = 20$ns. The required speedup when $N = 10000$ and $n = 16$ is always less than 2. So, the switch performance improves as the number of ports per SE increases. The performance can be improved by decreasing the cell duration time which increases the implementation complexity. Speedups are further improved when the counters are desynchronized. The satisfactory performance is achieved for the number of ports that rapidly decreases with n and so the switch connectivity degrades as n decreases.

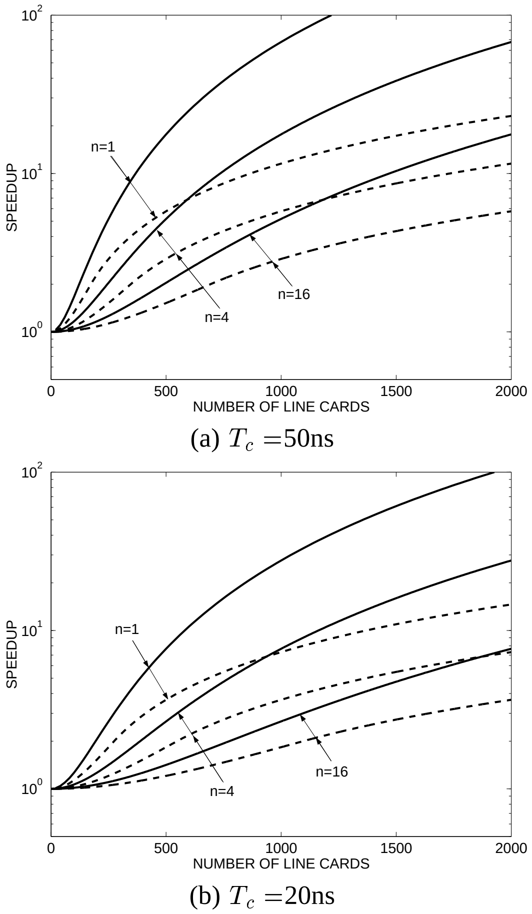
It is difficult to implement the high speed ports. Let us investigate if it would be worthwhile to make efforts to develop such ports. Assume that one SE is placed on one line-card, and line-cards have the specified capacity regardless on the number of ports per SE. Figure 8 shows the fabric utilization decrease with the number of line cards for various numbers of ports per SE, while Figure 9 shows the fabric speedup increase with the number of line cards for various numbers of ports per SE. Again, the solid curves represent the algorithms with independent counters, while the dashed curves correspond to the algorithms in which the counters are desynchronized. We can see that for a specified switch capacity, and therefore the number of line cards, the utilization of the fabric increases as the number of ports per line card increases. Or, the speeedup required for 100$\%$ utilization of the switch capacity decreases as the number of ports per line card



(a) T_c =50ns

(b) T_c =20ns

Fig. 8. Switch utilization: solid curves represent the performance with independent counters; dashed curves correspond to the performance with desynchronized counters.



(a) T_c =50ns

(b) T_c =20ns

Fig. 9. Fabric speedup: solid curves represent the performance with independent counters; dashed curves correspond to the performance with desynchronized counters.

increases. So, the smaller number of line cards are required to provide the same capacity if the number of ports per line card is larger. The larger number of ports also provides the richer connectivity, and the development of high-capacity ports does not seem advantageous.

Note that assuming different load balancing algorithms in PPS, the packet delay has been analyzed in [4] to reach $2N \cdot M_{max} \cdot T_c$, where M_{max} is the maximum number of multicast sessions per port. So, the delay in this switch can be very large $2N \cdot F \cdot T_c = 2N \cdot R \cdot T_c/G$, where G is the granularity of multicast sessions. In a switch with 10Tbps capacity, for the multicast session bandwidth granularity of 10Mbps, and $T_c = 50$ns, the packet delay of $D = 100$ms is unacceptably large.

In summary, the switch performance improves as the number of balanced flows decreases. The algorithms for which $N_f \leq N$ will perform well for all practical switch sizes in the case $n = m = l$. In [27], it was proposed that the end-to-end sessions are independently balanced in a switch. In that case $N_f \geq nN$, and consequently the performance is poorer than of the algorithm where a flow comprises cells from some input to some output. The performance of the latter protocol was not satisfactory in terms of the fabric utilization and the speedup required for the 100% switch utilization. On the other side, the algorithm in which a flow comprises cells bound from some input to some output SE can be implemented if SEs are crossbars, and it performs well for the practical switch sizes. The performance is even better when input SEs balance the traffic because the number of flows is decreased. However, the implementation of the algorithms where input SEs balance the traffic may be more complex, and, consequently, less scalable. First, the counters of the arbiter should be updated n times per cell time slot, which may require advanced processing capability, and may limit the number of SE ports i.e. the total switch capacity. Also, these algorithms assume the SEs with the shared buffers whose capacity was recognized to be smaller than the capacity of the cross-bar SEs. But, the performance of a Clos packet switch comprising the large number of the limited capacity SEs with shared buffers was shown to be also satisfactory when the number of ports per SE is sufficiently large. Finally, the switch performance is significantly improved when the counters are desynchronized. At the expense of a minor increase of the algorithm complexity, the fabric utilization is significantly increased. As a result, the switch consuming the smaller space and power will be able to provide the given high capacity.

VI. Conclusion

Clos packet switches provide high capacity. Clos packet switches are non-blocking when the load bound for either outputs or output SEs is balanced across the SEs in the middle stage. As a result, no centralized admission control is required in the described architecture. When reserving bandwidth for a new session, an input port has only to check if the appropriate output port has sufficient capacity. If the sufficient bandwidth is provisioned to users, admission control can be moved to the edge of the network, and a user would check if the destination user has enough capacity to receive data, and send the information accordingly. This distributed admission control would further enhance the dynamics of the Internet.

We investigated the fabric utilization under which the delay requirements of sensitive applications are met in Clos packet switches based on load balancing. We calculated the utilization for various load balancing algorithm in terms of the number of flows that are balanced, and various tolerable delays. The presented analysis is accurate, using minor approximations. As expected, the performance degrades as the number of balanced flows increases. The utilization was shown to be poor in the large switches in which end-to-end sessions are balanced independently. However, balancing the small number of flows readily provides required rate and delay guarantees in arbitrarily large switches. We showed that the performance is satisfactory in very large switches (with 4000 ports) when the cross-bars are used, if a flow comprises cells for the same output switching element. We also examined the performance of Clos packet switches using limited size shared buffers for switching elements. It was shown that their performance is satisfactory for equally large capacities, when the sufficiently large number of ports per switching element are deployed. The counter desynchronization slightly increases the algorithm complexity, but significantly improves the fabric utilization, and so it reduces the space and the power that the switch requires.

REFERENCES

[1] T. E. Anderson, S. S. Owicki, J. B. Saxe, and C. P. Thacker, “High-speed switch scheduling for local-area networks,” *ACM Transactions on Computer Systems,* vol. 11, no. 4, November 1993, pp. 319-352.

[2] T. Chaney, J. A. Fingerhut, M. Flucke, J. S. Turner, “Design of a gigabit ATM switch,” *Proceedings of INFOCOM 1997,* vol. 1, pp. 2-11.

[3] C. S. Chang, D. S. Lee and C. M. Lien, “Load balanced Birkhoff-von Neumann switches, Part II: multi-stage buffering,” *Computer Communications,* vol. 25, pp. 623-634, 2002.

[4] C. S. Chang, D. S. Lee and C. Y. Yue, “Providing guaranteed rate service in the load balanced Birkhoff-von Neumann switches,” *Proceedings of INFOCOM 2003.*

[5] H. J. Chao, “Saturn: A terabit packet switch using dual round-robin,” *Proceedings of GLOBECOM 2000,* pp. 487-495.

[6] C. Clos, “A study of non-blocking switching networks,” *Bell Systems Technology Journal,* vol. 32, 1953, pp. 406-424.

[7] J. Hui, *Switching and Traffic Theory for Integrated Broadband Networks,* Kluwer Academic Press 1990.

[8] F. K. Hwang, *The mathematical theory of nonblocking switching networks,* World Scientific, 1998.

[9] S. Iyer, and N. McKeown, “Analysis of the Parallel Packet Switch architecture,” *IEEE/ACM Transactions on Networking,* vol. 11, no. 2, April 2003, pp. 314-324.

[10] I. Keslassy, S. T. Chuang, K. Yu, D. Miller, M. Horowitz, O. Solgaard, N. McKeown, “Scaling Internet routers using optics,” *Proceedings of ACM SIGCOMM 2003.*

[11] T. McDermott, and T. Brewer, “Large-scale IP router using a high-speed optical switch element,” *OSA Journal on Optical Networking, www.osa-jon.org,* July 2003, pp. 229-241.

[12] W. Kabacinski, C. T. Lea, G. Xue, “50th anniversary of Clos networks,” *IEEE Communication Magazine*, vol. 41, no. 10, October 2003, pp. 26-64.

[13] N. McKeown *et al.*, “The Tiny Tera: A packet switch core,” *IEEE Micro*, vol. 17, no. 1, Jan.-Feb. 1997, pp. 26-33.

[14] E. Oki, Z. Jing, R. Rojas-Cessa, H. J. Chao, “Concurrent round-robin-based dispatching schemes for Clos-network switches,” *IEEE/ACM Transactions on Networking,* vol. 10, no. 6, December 2002, pp. 830-844.

[15] A. Smiljanić, “Flexible bandwidth allocation in terabit packet switches,” *Proceedings of IEEE Conference on High Performance Switching and Routing,* June 2000, pp. 233-241.

[16] A. Smiljanić, “Flexible bandwidth allocation in high-capacity packet switches,” *IEEE/ACM Transactions on Networking,* April 2002, pp. 287-293.

[17] A. Smiljanić, “Scheduling of multicast traffic in high-capacity packet switches,” *IEICE/IEEE Workshop on High-Performance Switching and Routing,* May 2002, pp. 29-33.

[18] A. Smiljanić, “Scheduling of multicast traffic in high-capacity packet switches,” *IEEE Communication Magazine,* November 2002, pp. 72-77.

[19] A. Smiljanić, “Bandwidth Reservations by Maximal Matching Algorithms,” *IEEE Communication Letters,* March 2004, pp. 177-179.

[20] A. Smiljanić, “Performance of load balancing algorithms in Clos packet switches,” *Proceedings of IEEE Workshop on High Performance Switching and Routing,* April 2004, pp. 304-308.

[21] A. Smiljanić, “Performance of load balancing algorithms in Clos packet switches,” *Invited Presentation at Stanford Workshop on Load-Balancing,* May 2004.

[22] A. Smiljanić, “Load balancing algorithms in Clos packet switches,” *Proceedings of IEEE International Conference on Communications,* June 2004.

[23] A. Smiljanić, “High performance routers,” *invited paper at joint Opto-electronic and Communication Conference and International Conference on Optical Internet,* Yokohama, Japan, July 2004.

[24] A. Smiljanić, “Terabit switching algorithms,” *invited paper at SPIE Asian Pacific Optical Communication Conference,* Beijing, China, November 2004.

[25] A. Smiljanić, and Miloš Petrović, “Speedup of Clos packet switches that provide delay guarantees, *IEEE Workshop on High Performance Switching and Routing*, Hong Kong, China, May 2005.

[26] Y. Tamir, and H. C. Chi, “Symmetric crossbar arbiters for VLSI communication switches,” *IEEE Transactions on Parallel and Distributed Systems,* vol. 4, no. 1, 1993, pp. 13-27.

[27] J. S. Turner, “An optimal nonblocking multicast virtual circuit switch,” *Proceeding of INFOCOM 1994,* vol. 1, pp. 298-305.

Aleksandra Smiljanić (M ’96) received the M.A. and Ph.D. degrees in electrical engineering from Princeton University in 1996 and 1999, respectively. She got the B.Sc. degree in electrical engineering at Belgrade University in 1993. Her area of research is high performance switching and routing.

Currently, Aleksandra works as a Professor at Belgrade University in Serbia, as an Associate Research Professor at Brooklyn Polytechnic University and an Adjunct Professor at Stony Brook University in New York. She had worked for AT&T Labs Research from 1999 until 2004. She worked for two summers at NEC USA on a design of the packet switch with terabit capacity.

Aleksandra Smiljanić is the author of numerous conference and journal papers in the area of high performance switching and routing. She is the inventor of seven patents, and of two patent applications. Aleksandra Smiljanić is the author of the Best Papers at IEEE Conference on High Performance Switching and Routing 2000, and IEICE/IEEE Workshop on High Performance Switching and Routing 2002. She got the Research Excellence Award at AT&T Labs in 2000. She is a recipient of the Aleksandar Damjanović Prize as the best student in her class at Belgrade University, 1993. Before university, she won numerous prizes in Yugoslav and international competitions in mathematics and physics. Aleksandra Smiljanić is an Editor of IEEE Communication Letters and of OSA Journal on Optical Networking.