Configuration of Cycle Duration in Cyclic Queuing and Forwarding

Damien Guidolin–Pina, Marc Boyer, Jean-Yves Le Boudec

Configuration of Cycle Duration in Cyclic Queuing and Forwarding

Damien Guidolin--Pina, Marc Boyer, and Jean-Yves Le Boudec, *Fellow, IEEE,*

Abstract—Cyclic Queuing and Forwarding (CQF) is a mechanism defined by IEEE TSN for providing low jitter in a deterministic network. CQF uses a common time cycle and two buffers per node output port. During one cycle, incoming packets are stored in one buffer while packets in the other buffer are being transmitted; at the end of a cycle, the roles of the two buffers are exchanged. CQF provides very simple bounds on latency and jitter. Its correct operation requires a large enough cycle duration so that all packets received by a node in one cycle can be forwarded during the next cycle. We give a necessary and sufficient condition for this to hold. Our condition depends on the link properties and the flow characteristics at the network input. We apply the condition to obtain the minimal admissible cycle duration. We observe that the minimal cycle duration is not always margin-safe, i.e., larger values might be non-admissible, which suggests replacing the minimal cycle duration with the minimal margin-safe cycle duration.

Index Terms—TSN; CQF; Peristaltic Shaper; Cyclic Queuing and Forwarding.

1 Introduction

To offer a standard real-time data network, the IEEE Time Sensitive Networking working group has defined several extensions to Ethernet. Among others, Cyclic Queuing and Forwarding (CQF), inspired by the "stop-and-go" queuing [1], has been defined in order to offer guaranteed delay and limited jitter [2]. In short (see Section 2 for more details), CQF considers a common time cycle, T, and uses two queues per node output port to alternatively store or forward packets. It guarantees that the delay experienced by a packet crossing h nodes along its path is between $(h-1)T$ and $(h+1)T$.

CQF can be used by time-triggered flows whose emission times are statically known in advance for every packet [3], [4], or by asynchronous flows, whose emission times are not known in advance. In this paper, we consider a CQF network used by asynchronous flows. We allow arbitrary link speeds between nodes. Asynchronous flows are assumed to be constrained by arrival curves (e.g. the number of bits emitted during τ seconds does not exceed $r\tau + b$, where r is the rate limit and b the allowed burst) [5].

CQF has appealing characteristics: it is simple to implement, offers a simple expression of the latency bound, provides a limited delay jitter ($2T$) [6] and can also easily handle cyclic dependencies (whereas other real-time mechanisms may require some attention in such cases, [7]–[10]). However, the correct operation of CQF requires that the following two properties hold:

- *time alignment*: all packets sent by a node in one cycle are received by the next downstream node in one cycle (and not spread over several consecutive cycles);
- *large enough cycle*: all packets received by a node in one cycle can be forwarded during the next cycle.

To achieve the former property, a guard band is used at the beginning and end of every cycle. The minimum value of the guard band depends on the cycle offsets, the transmission and propagation times, and the clock stability properties. It can be computed using the methods in [11]. In this paper, we assume that the guard band and the offsets are computed such that the time-alignment condition holds.

The latter property, which is the focus of this paper, depends on the duration of the cycle and on the traffic. Obviously, the sum of the rates of all flows traversing a link must be less than the link speed, but this may not be sufficient, due to burstiness. A large cycle duration can accommodate more bursty sources but leads to larger latency and jitter. Our main result, in Theorem 1, is a necessary and sufficient condition for the large-enough-cycle condition to hold. It involves the link characteristics and the arrival curves of flows (i.e. rates and bursts) at the network input. Remarkably, it is only the value of bursts at the network input that matters, whereas in general, the burstiness of flows increases due to multiplexing inside a network. Our analysis reveals that CQF limits such a burstiness increase to a timescale less than the cycle duration, which explains its attractive low-jitter property. Our result incorporates the effect of clock non-idealities.

Then we apply Theorem 1 to compute the minimal cycle duration for a given configuration of links and flows. Minimizing the cycle duration is of primary interest as it determines the end-to-end delays and jitters. Surprisingly, we find that the set of admissible cycle durations (i.e., that guarantees the large-enough-cycle property) is not an interval. Specifically, increasing the value of an admissible cycle duration may make it non-admissible, which may be a problem in practice as it is common to add a safety margin. This motivates the definition of the smallest admissible value such that any larger value is also admissible, which we

- *Damien Guidolin--Pina is with RealTime-at-Work, 54000 Nancy, France.*
- *Marc Boyer is with ONERA/DTIS, Université de Toulouse – F-31055 Toulouse, France.*
- *Jean-Yves Le Boudec is with EPFL, 1015 Lausanne, Suisse.*

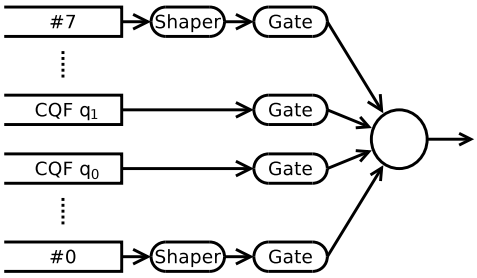


Fig. 1: Queues of a bridge using CQF policy.

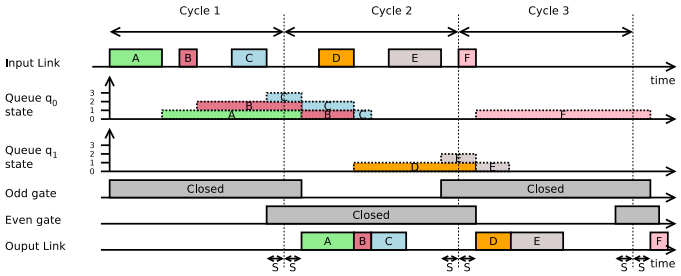


Fig. 2: Local behaviour of a CQF node.

call "margin-safe cycle duration". Numerical experiments indicate that the margin-safe cycle duration may be twice as large as the minimal cycle duration when the system load exceeds 60%.

The rest of the paper is organized as follows. Section 2 gives a description of CQF, the system assumptions, and the notation. Section 3 gives our main result, namely, a practical condition for the large-enough-cycle condition to hold. In Section 4, we apply the main result to the computation of the minimal cycle duration and show in some examples that it may not be margin-safe. Section 5 provides a review of the state of the art and Section 6 concludes the paper.

2 Presentation of CQF

The standard documents that describe CQF use the term "frame" for "packet", as is typical for layer-2 standards. In the scientific literature, it is more common to use "packet", and we follow this convention.

2.1 CQF in isolation

Per port behaviour

Consider a TSN output port, as represented in Figure 1, with two queues, called q_0 and q_1, dedicated to a set of CQF flows. Theses two queues represents a single CQF class on this port (for other TSN mechanisms, a traffic class corresponds to a single queue, but CQF requires two). The CQF node has a periodic behaviour, with period $T > 0$ (called "cycle time"), assumed to be the same for all CQF nodes in the network. Each node tags each interval as even or odd. The packets received during an even cycle are stored in the queue q_1, whose gate is closed, meaning that the packets are not forwarded. During this even cycle, the gate of the queue q_0 is open, and the packets compete with other queues for transmission (using a non-preemptive arbitration policy). During an odd cycle, the converse behaviour occurs. A guard band, of duration S, forbids any emission at the start and the end of each cycle.

This behaviour is illustrated in Figure 2. Packets A, B, C are received during cycle 1, an odd cycle; they are stored in the queue q_0 and the output gate of this queue is closed. As soon as this gate is open (at the start of cycle 2, plus the guard band), packets A, B, C are forwarded on the output link. During this cycle 2, the received packets D and E are stored in the queue q_1, and forwarded in cycle 3.

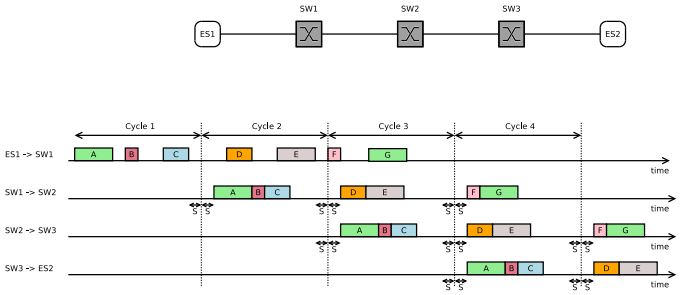


Fig. 3: Behaviour of a sequence of CQF nodes.

Global behaviour

When several nodes are involved, CQF requires a time-alignment condition, which expresses that the cycle boundaries of nodes are enough aligned such that all packets sent by a node in one cycle are received by the next downstream node in one cycle. CQF requires all clocks to be synchronized, but the cycle start times need not be exactly the same at every node, and may differ by some offsets, not shown on the figures. The time alignment condition requires the guard band to be large enough; the minimal guard band duration depends on the offsets and the quality of time synchronization: it can be computed using the method in [11]. In this paper, we assume that this time alignment condition holds. The global behaviour of CQF network can then be illustrated as in Figure 3 (where the queue and gate states have been omitted). Here, packets A, B and D emitted by the end-system in cycle 1 are forwarded by the switch SW1 at cycle 2, then forwarded by the switch SW2 at cycle 3, etc.

For correct operation, in addition to the time-alignment condition, CQF also requires that cycles are large enough to be able to empty one buffer in one cycle. Figure 4 illustrates the issue, by showing a case where this latter condition does not hold. Here, too many packets are received in cycle 1, so that not all packets received in cycle 1 can be transmitted in cycle 2 (this is the case for packet C, the transmission of which has to be postponed to cycle 4). Note that if the bitrate of the link from SW to ES3 were 11/9 larger than the one of the incoming links, such a problem would not occur.

The goal of this paper is to establish the conditions for the cycle-time to be large enough. When this and the time-alignment condition hold, CQF guarantees that if a flow follows a path made of h hops, its end-to-end delay is in the range $[(h-1)T, (h+1)T]$ and its delay-jitter is upper bounded by $2T$ [12]. For example, any packet in Figure 3 experiments a delay between $2T$ and $4T$.

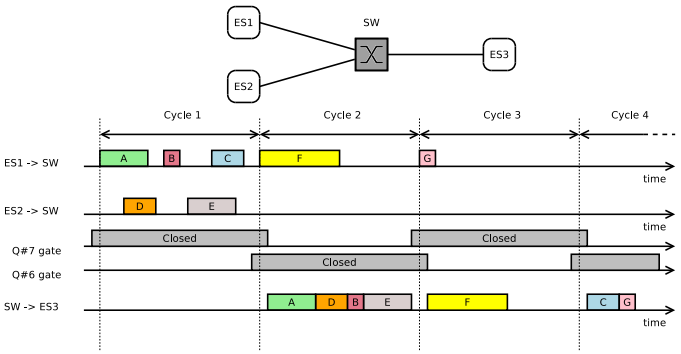


Fig. 4: Example of cycle time not large enough.

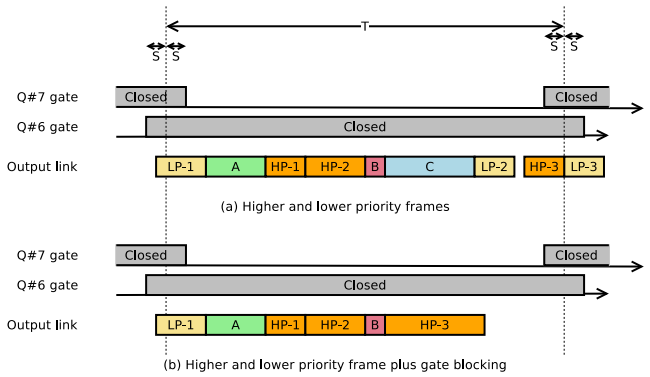


Fig. 5: Impact of CBS integration with other priorities, without preemption (assuming that CQF packets A, B and C have been received during the previous cycle).

2.2 CQF, interference and other classes

One strength of CQF is that it may host several kinds of traffic. Whereas the previous section was presenting CQF in isolation, this section shows hows CQF is inserted in a TSN output port.

As depicted in Figure 1, a CQF class may share the output port with other queues. There are up to 8 queues in a TSN output port, formally called "traffic classes", numbered from #0 to #7.

The arbitration policy is a static priority (class #7 having the highest priority). A CQF class can be set to any priority level. We may have $q_1 = \#7$ and $q_0 = \#6$ to set CQF to the highest priority, but we may also have $q_1 = \#6$ and $q_0 = \#5$ and leave class #7 to some higher priority flows. The system depicted in Figure 1 represents a system where $q_1 = \#n$, $q_1 = \#(n-1)$ with some n such that $7 > n > 1$. In this paper, we assume only that the CQF queues are adjacent *i.e.*, the difference between the priority level of the two CQF queues is always equal to one. Also note that $q_1 = \#7$ and $q_0 = \#6$ and $q_1 = \#6$ and $q_0 = \#7$ give similar behaviour[1]. The CQF packets may have to compete with higher priority flows when their own gate is open[2] and also with one lower priority packet.

Consider first a system without preemption. Higher priority packets may compete with CQF at any time when CQF gate is open, even if they have to wait the end of a CQF packet to access to the output link. Lower priority packet may also compete at any time, but since they are of lower priority, they will lose the arbitration when the CQF gate is open. Note that one lower priority packet may start its emission before the gate opening, and keep the link up to end of emission (as illustrated by packet LP-1 in Figure 5-(a)). Then, the blocking related to lower priority packet is upper bounded by the maximal size of a lower priority packet (while any upper bound on the blocking due to higher priority packet must be provided by the network designer[3]). Figure 5-(a) also shows that, once all CQF packets received in the previous cycle have been forwarded, lower priority packets can be sent (cf. packet LP-2).

Another effect may prevent CQF from using the full gate opening time: the next-closing-event blocking. In TSN, if a packet cannot be fully emitted before the next closing of its own gate, it will not be sent. Then, considering the example of Figure 5-(b), if the last CQF packet C is too large w.r.t the next closing event, the remaining part of the cycle cannot be used by CQF. Note that, in our interpretation of the standard, one lower priority packet can be sent during this time if its own gate is open during enough time (even if it may imply to do such arbitration in a very short time), but such a situation is not represented in Figure 5-(b) for sake of simplicity. As will be shown in Section 3, in a correctly configured CQF network[4], CQF packets will not suffer the next-closing-event blocking effect associated to the end of the cycle.

A preemption mechanisms is introduced in [15], [16]. The transmission of a *preemptable* packet can be interrupted if an *express* packet is ready for transmission; if it remains at least 144 bytes of the preemptable packet to be emitted [17], the transmission of the preemptable packet is stopped and the transmission of the express packet starts. When the preemptable packet is again eligible for transmission, it can resume the transmission but an overhead is added. Note that there is one single level of preemption: an express packet that has preempted a preemptable packet cannot be preempted by another express packet, regardless of their relative priorities.

If the queues of priority lower than the two CQF queues are preemptable, and if the CQF queues are express, the blocking due to lower priority packet is reduced to 143 bytes. The associated overhead (20 bytes, added at packet restart, denoted "O" in Figure 6) will not compete with CQF in the current cycle (it will be sent when CQF does no more require access to the link). But in this case, the higher priority packets will not be able to preempt CQF packets. This situation is illustrated in Figure 6-(a): the transmission of LP-1 is interrupted by the transmission of A, and restarts after the transmission of C.

1. The standard name the queues used by CQF either "queue 1" and "queue 2" when presenting the behaviour [13, § T.2], but also uses the expressions "queue 7" and "queue 6" when entering into implementation details [13, § T.4]. To avoid this ambiguity, we have introduced notations q_0 and q_1.

2. Note that there may exist several CQF classes of different priorities, then, a CQF class may have to compete with a higher priority CQF class [5].

3. It may be also partially deduced from the configuration of the policing elements [14].

4. A more formal condition on the configuration will be given in Section 3.

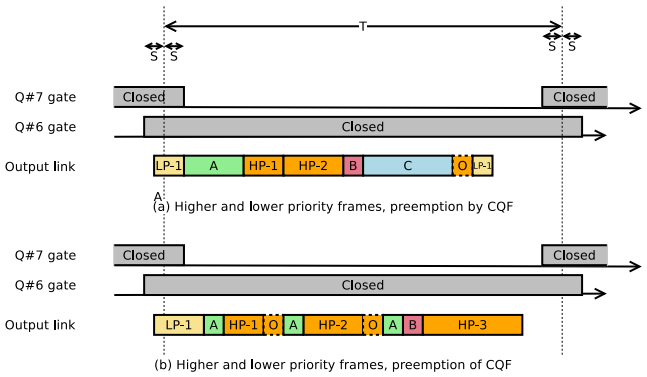


Fig. 6: Impact of CBS integration with other priorities, with preemption (assuming that CQF packets A, B and C have been received during the previous cycle). The relative size of the packets and the overhead does not fit the reality.

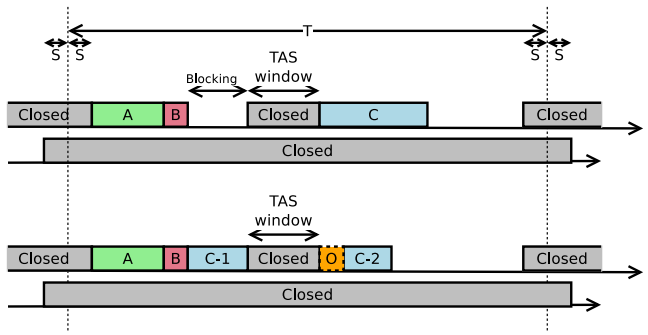


Fig. 7: Impact of TAS/CBS integration, with or without preemption of packet C. (assuming that CQF packets A, B and C have been received during the previous cycle). The relative size of the packets and the overhead does not fit the reality.

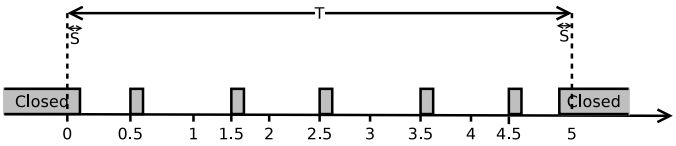


Fig. 8: A CQF cycle of 5ms with 5 gate closing (dedicated to TAS) – units are in milliseconds.

If the CQF queue is preemptable, and if some higher priority queue is express, the CQF packets can be preempted, creating some overhead that will use part of the bandwidth in the current cycle. Note that even if a CQF packets can be preempted several times, one high priority packet can create at most one preemption overhead. Then the number of overheads in one cycle is upper bounded by the number of higher priority packets in this cycle.

CQF may also interfere with Scheduled Traffic, implemented using a Time Aware Shaper (TAS, [18], [19]), that will require an exclusive access to the output port, and close the gate of the CQF packet during the transmission of a Time-Triggered packet (also called scheduled packet), even in an open cycle. Such a gate closing of duration d will have a worst-case impact larger than d: due to the next-closing-event blocking, one may loose one full packet size per TAS window in the CQF cycle, as illustrated in Figure 7-(a). Note that [20] propose to use (and extend) the guard band to send the scheduled traffic to avoid this effect.

One may also consider that preemption may be used to reduce this blocking effect, as illustrated in Figure 7-(b), but our understanding of standard is that a closing event does not trigger a preemption (only express packets do).

One may wonder if, at gate re-opening, CQF may suffer for a lower priority blocking, as at start of CQF cycle. In the common TAS implementation, using exclusive gating, this is not the case. With exclusive gating, during a TAS window, all other gates are closed, then all these gates re-open at the same time, and CQF will win any arbitration with lower priority packets.

In summary, during an open cycle, CQF may be blocked by:

- one single lower priority packet (the amount of blocking being a full packet if there is no preemption, and if there is preemption and if the CQF queue is express and the lower priority queues are preemptable, this blocking is reduced to 143 bytes);
- some non scheduled higher priority flow;
- some preemption overhead, if there is preemption and the CQF queue is preemptable and the higher priority flows are express (the amount of overhead can be a bounded number of higher priority packet);
- several closed windows (due to TAS), each window of duration d leading to a loss of $Rd+163\times 8$ pseudo-bits.

We let $Bl_{j,k}$ denote the amount of blocking due to all these effects, during cycle k of the CQF port j. Note that $Bl_{j,k}$ is either 0 or not less than 72 bytes (minimal size of and Ethernet packet plus the preamble). In addition, we let $\overline{Bl}_j$ denote an upper bound on the amount of blocking due to all these effects on the CQF port j i.e., for all $k \in \mathbb{N}$,

$$\overline{Bl}_j \geq Bl_{j,k}. \quad (1) \quad \texttt{\{eq:alphaHP\}}$$

The bound $\overline{Bl}_j$ is *reachable* if there exists a behaviour of the system and a cycle k such that $\overline{Bl}_j = Bl_{j,k}$.

The bound $\overline{Bl}_j$ must not consider the blocking due to the next-closing-event associated with the end of the cycle because, as mentioned above, in a correctly configured CQF, this does not affect CQF packets.

In the rest of this section we illustrate the computation of $\overline{Bl}_j$ on a few examples.

Consider a port j, with some for Scheduled Traffic in class #7, some high-priority traffic in class #6, and a CQF class using queues #5 and #4 and some other traffic in lower-priority queues #3 to #0 on a 1Gb/s output link. The CQF cycle time is 5ms. The TAS mechanisms reserves for the scheduled traffic one window of 0.1ms every millisecond (using 10% of the bandwidth) with an offset of 0.5ms relative to the start of the cycle time. The traffic in class #6 uses no more than 15% of the bandwidth in each cycle. The cycle with gate closing is depicted in Figure 8.

If the system is without preemption, a bound $\overline{Bl}_j$ can be

computed as

$$\overbrace{1542 \times 8b}^{\text{one low priority packet}} + \overbrace{5ms \times 1Gb/s \times \frac{15}{100}}^{\text{class \#6 usage}} + \underbrace{5 \times (0.1ms \times 1Gb/s + 168 \times 8b)}_{\text{5 closed windows}}. \quad (2)$$

If the CQF class can preempt the lower priority traffic (if the preemption is on, and the classes $\#6$, $\#5$ and $\#4$ are express and the classes $\#3$ to $\#0$ are preemptable), then a bound $\overline{Bl}_j$ can be computed as

$$\overbrace{143 \times 8b}^{\text{one low priority non-preemptable packet}} + \overbrace{5ms \times 1Gb/s \times \frac{15}{100}}^{\text{class \#6 usage}} + \underbrace{5 \times (0.1ms \times 1Gb/s + 168 \times 8b)}_{\text{5 closed windows}}. \quad (3)$$

If the CQF class can be preempted by the higher priority traffic (if the preemption is on, and the classes $\#6$ is express, and the classes $\#5$ to $\#0$ are preemptable), then, the preemption overhead must be taken into account, leading to the following value for $\overline{Bl}_j$:

$$\overbrace{143 \times 8b}^{\text{one low priority non-preemptable packet}} + \overbrace{5ms \times 1Gb/s \times \frac{15}{100}}^{\text{class \#6 usage}} + \underbrace{5 \times (0.1ms \times 1Gb/s + 168 \times 8b)}_{\text{5 closed windows}} + \underbrace{n \times 20b}_{\text{preemption overhead}}, \quad (4)$$

where n is an upper bound on the number of packets of class $\#6$ in each cycle (if a minimal size on the packet size $\underline{L}_{\#6}$ is known, then $n \leq \frac{5ms \times 1Gb/s \times \frac{15}{100}}{\underline{L}_{\#6}}$, but depending on the context, there may be other ways to get an upper bound, based on the stream traffic contract or more generic approaches [21]).

2.3 Time Synchronisation and Clock Nonidealities and Arrival Curves.

CQF requires that network nodes are time synchronized, using for example [22]. For simplicity, in Figures 3 and 4, cycles of different nodes are drawn perfectly aligned, but in reality, perfect time synchronization does not exist; furthermore, transmission and propagation delays should be accounted for and there may be a different time offset at every node (this may mitigate the time alignement issue [11]). The role of the guard band is to avoid all time alignment problems.

At the microsecond time scale, clock nonidealities also have to be taken into account. In the context of a time-sensitive network, clock non ideality is captured by three network-wide parameters: the clock stability bound $\rho \geq 1$, the timing jitter bound $\eta \geq 0$ and the synchronization error bound $\Delta \geq 0$ [23]. In a TSN network synchronized with gPTP (generic PTP), the reference values are $\rho = 1.0001$ [22, Annex B.1.1], $\eta = 2$ns [22, Annex B.1.3.1] and $\Delta = 1\mu$s [22, Section B.3]. These parameters provide error bounds for the difference in measurements of one same interval performed with two different clocks; specifically, if one clock measures a duration d and the other one a duration d', then

{eq-clk} $$\max\left(d - 2\Delta, \frac{d-\eta}{\rho}\right) \leq d' \leq \min(d + 2\Delta, \rho d + \eta). \quad (5)$$

2.4 System Model and Notations

The notation used throughout the paper is listed in Table 1. The system of interest is a network and one CQF class; if the network has several CQF classes, the other CQF classes are considered as higher or lower priority classes with respect to the CQF class of interest. The network is made of switches and end-systems.

Flows of data are generated at end-systems. The number of bits generated by flow i at its source end-system is constrained by an arrival curve α^i, i.e., $\alpha^i(d)$ is an upper bound on the number of bits generated by flow i on any interval of duration d, including any overhead like preamble and inter-frame gap (this is called an arrival curve in network calculus). For example, a flow i that is constrained to emit at most L bits every τ seconds is characterized by the arrival curve $\alpha^i(d) = L\left\lceil \frac{d}{\tau} \right\rceil$ where $\lceil \cdot \rceil$ denotes the ceiling function. Such an arrival curve provides the most accurate delay and backlog bounds [24], but, for tractability, is often replaced by a linear upper bound (called leacky bucket function) such as $\alpha^i(d) = rd + b$ where $b = L$ is called the burst parameter and $r = \frac{L}{\tau}$ the rate parameter.

The arrival curve α^i expresses quantities that are implemented using the local clock of the source. When another system, such as a switch output port, needs to make assumptions about the number of bits generated by the source during a time interval measured with this system's clock (not the source's clock), then the arrival curve needs to be inflated in order to account for clock non-ideality [23]. The arrival curve for flow i that can be assumed by a system that is not the source of flow i is derived from (5) and is given by

{eq:arrival_c} $$\tilde{\alpha}^i(d) = \alpha^i\left(\min(d + 2\Delta, \rho d + \eta)\right). \quad (6)$$

In other words, the number of bits generated by source i during any time interval of duration d, as measured by the local clock of a switch, is upper bounded by $\tilde{\alpha}^i(d)$. For example, if $\alpha^i(d) = rd + b$ (leaky bucket function), then

{eq-jksd} $$\tilde{\alpha}^i(d) = \min\left(rd + 2r\Delta + b, r\rho d + r\eta + b\right). \quad (7)$$

Let $\mathcal{P}$ denote the set of CQF output ports in the network; note that the end-system output ports typically do not implement CQF, and thus $\mathcal{P}$ only includes output ports of switches, not of sources. The duration of the CQF cycle is T, and there is a guard band of duration S at the beginning and at the end of every cycle. These values are identical at all ports. In contrast, the links may have different bit rates. Port j uses a time offset o_j (with $0 \leq o_j < T$), at which the initial cycle begins. And the kth cycle starts at $o_j + kT$ (up to clock nonidealities).

Finally, let $\mathrm{Fl}(j)$ denote the set of flows crossing the CQF port j.

3 Condition on the Cycle Time

In this section we give a necessary and sufficient criterion for the large-enough-cycle (LEC) condition to hold.

Theorem 1 (Global LEC Condition)**.** *Assume that the guard band and the offsets are such that time alignment holds [11]. Let*

TABLE 1: Notations.

$\mathcal{P}$	Set of CQF ports in the network
R_j	Bit rate of port j
Δ	Clock synchronization error bound (e.g. $\Delta = 1\mu$s)
ρ	Clock stability bound (e.g. $\rho = 1.0001$)
η	Clock timing jitter bound (e.g. $\eta = 2$ns)
T	Duration of cycle
S	Guard band at begin and end of a cycle
o_j	Cycle offset at port j
$\lfloor x \rfloor$	Floor of the real number x
$n\%m$	Remainder modulo m of the integer n
$\text{In}_j^i(k)$	Number of bits of flow i that arrive at port j in cycle k
$\text{Out}_j^i(k)$	Number of bits of flow i leaving port j in cycle k
$\text{Out}_j^{HP}(k)$	Amount of higher priority data competing for the access to port j during cycle k
$\text{Fl}(j)$	Set of flows crossing port j
$\overline{Bl}_j$	Upper bound on the amount of blocking due to other classes on the CQF port j (Section 2.2)
$\alpha^i(d)$	Upper bound on the number of bits generated by flow i on any interval of duration d measured with the source's clock
$\tilde{\alpha}^i(d)$	Upper bound on the number of bits generated by flow i on any interval of duration d measured with the clock of a switch
$\mathcal{T}_j$	Set of T respecting eq. (8) on port p_j
T^{opt}	Global minimal cycle time
T^{safe}	Global minimal margin safe cycle time
T^{conc}	Maximum of local concave cycle time

$T > 0$, it satisfies the Large Enough Cycle condition if for every port $j \in \mathcal{P}$

$$\sum_{i \in Fl(j)} \tilde{\alpha}^i(T) \leq R_j(T - 2S) - \overline{Bl}_j. \tag{8}$$

Conversely, if $\overline{Bl}_j$ is reachable for every port j (see Equation (1) and its comments), this is a necessary condition.

The theorem gives a condition on the cycle time T that depends on the system load via the arrival curves at network input: the condition at port j does *not* depend on the number of hops between the sources of the flows and this port. Indeed,a remarkable property of CQF, which appears in the proof of the theorem, is that *the burstiness of flows measured at the duration of the cycle time, does not increase as flows progress through the network.* This is in stark contrast with other network scheduling techniques (such as FIFO per-class), where it is known that flow bursiness increases at every hop.

Figure 9 illustrates Theorem 1 with a simple situation, with only two CQF flows. Graphically, the solutions of Equation 8 are all the times T such that the sum of the arrival curves (the plain black curve) is below the service curve (the dotted red curve). This set of solution is shown in thick blue along the T axe and it will be defined in the following section, Section 4.1.1. Also, the specific times T_j^{opt} and T_j^{safe} will also be presented in Section 4.1.1.

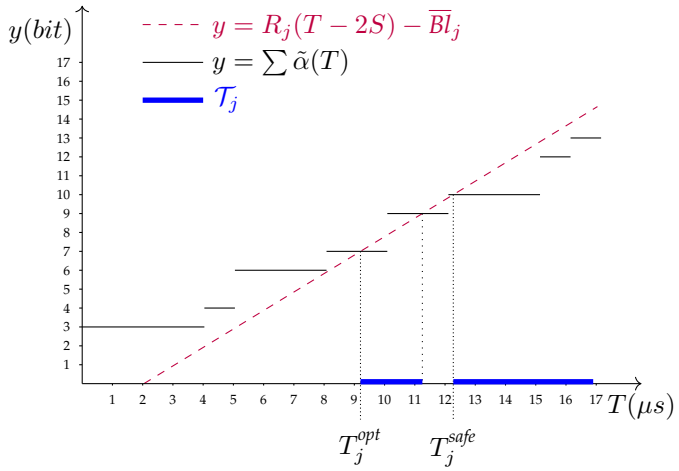


Fig. 9: Illustration of $\mathcal{T}_j$, $\alpha^1(d) = \lceil \frac{d}{4} \rceil$, $\alpha^2(d) = 2\lceil \frac{d}{5} \rceil$, $R_j = 1bit/\mu s$, $S = \frac{T}{100}$, $(\rho, \eta, \Delta) = (\frac{100}{99}, 0, 0)$, $\overline{Bl}_j = 2bit$.

Sketch of proof.: The proof of the sufficient condition is as follows:

1) Let $B_{j,0}(k)$ denote the backlog in $q_{j,0}$ at the end of cycle k (and similarly for $B_{j,1}(k)$). Also let $\text{In}_j^i(k)$ denote the number of bits of flow i that arrive at port j during cycle k. We show the following property: for a given port j and a given cycle k, if there is no backlog at the end of the previous cycle ($B_{j,(k+1)\%2}(k-1) = 0$) and the amount of data received during the current cycle ($\sum_{i \in \text{Fl}(j)} \text{In}_j^i(k)$) is upper bounded by $R_j(T-2S) - \overline{Bl}_j$ then there is no backlog at the end of the current cycle ($B_{j,k\%2}(k) = 0$).
2) Let $l(i,j)$ denote the number of hops from the source of the flow i until port j ($l(i,j) = 1$ if port j is the first switch output port on the path of flow i). For two positive integers k, d, we say that the property $P(k,d)$ holds if and only if, for all $j \in \mathcal{P}$ and all $i \in \text{Fl}(j)$ such that $l(i,j) \leq d$, the following four properties hold:

$$\text{In}_j^i(k) \leq \tilde{\alpha}^i(T) \tag{P1}$$
$$B_{j,(k+1)\,\%\,2}(k-1) = 0 \tag{P2}$$
$$B_{j,(k+1)\,\%\,2}(k) \leq R_j(T - 2S) - \overline{Bl}_j \tag{P3}$$
$$\text{In}_j^i(k) = \text{Out}_j^i(k+1). \tag{P4}$$

We prove this property by a double induction on k and d. The property $P(k,d)$ shows that the burstiness of flows in CQF does not increase along the path of the flows and is bounded by what enters during a cycle of duration T (P1). Also it gives an upper bound on the buffer usage (P2 and P3). Finally, (P4) implies the LEC condition.

The necessary condition is proven by exhibiting a trajectory that achieves the bound.

The full proof is given in Appendix A.

4 Computation of the Cycle Time

Theorem 1 gives a condition on the cycle time T that depends on the arrival curves at network input. The next step is to use the condition to determine admissible values of T. Suprisingly, as we show now, if some value T satisfies

the condition in Theorem 1, there may exist some $T' > T$ that does not. This makes the problem of choosing a good T more complex than expected.

4.1 Local Conditions on Cycle Time

4.1.1 Definitions

Definition/Property 1 (Local, Minimum Cycle Time)**.** *Consider a port j. Define $\mathcal{T}_j$ as the set of values of the cycle time T that satisfy* (8) *for this value of j. Let $T_j^{opt} \stackrel{def}{=} \inf \mathcal{T}_j$. If $T_j^{opt} > 0$, then $T_j^{opt} \in \mathcal{T}_j$.*

In other words, T_j^{opt} is the smallest cycle time that satisfies the large-enough-cycle condition at port j. Because the arrival curves in (8) cannot generally be assumed to be continuous (only left-continuity is generally assumed, [25]), it is not obvious whether the infimum of $\mathcal{T}_j$ is in $\mathcal{T}_j$. The proof that this does hold is given in Appendix B.

Since the latency is directly related to the cycle time, it seems beneficial to chose the smallest value, and thus T_j^{opt} is, as far as latency is concerned, the optimal cycle time when considering the constraints at port j. However, as we see next, it may not be the most desirable value.

This is because, in general, we cannot expect $\mathcal{T}_j$ to be an interval. The set $\mathcal{T}_j$ is plotted as a thick line along the T axis on Figure 9. It is not an interval. In particular, the optimal cycle time is $T_j^{\text{opt}} = 9.18\mu s$, but the value $T = 12\mu s$ is not a correct cycle time (it does not belong to $\mathcal{T}_j$).

In practice, when we choose a cycle time, we may want to round it up in order to keep some margin in view of a future evolution of the system. Choosing T_j^{opt} is therefore not always a good option. This leads to the following definition:

Definition/Property 2 (Local, Margin-Safe Cycle Time)**.** *Consider a port j. We say that T satisfies the local margin-safe condition at port j if and only if, for all $t \geq T$, $t \in \mathcal{T}_j$. Let $\mathcal{T}_j^{safe}$ denote the set of values that satisfy the local margin-safe condition at port j and $T_j^{safe} \stackrel{def}{=} \inf \mathcal{T}_j^{safe}$. If $T_j^{safe} > 0$, then $\mathcal{T}_j^{safe} = \left[T_j^{safe}, +\infty\right)$.*

The property says that the set of values of T that satisfy the local margin-safe condition at port j is always a closed interval. The proof is given in Appendix C. By definition, every $T \geq T_j^{\text{safe}}$ satisfies the LEC condition at port j.

It follows immediately from the definitions that $T_j^{\text{safe}} \geq T_j^{\text{opt}}$, but in general we cannot expect that there is equality. For example, in Figure 9, $T_j^{\text{safe}} = 12.24\mu s$ whereas $T_j^{\text{opt}} = 9.18\mu s$.

4.1.2 Concave Arrival Curves

In the special case where all arrival curves are concave, the complexity that comes from the fact that $\mathcal{T}_j$ may not be an interval disappears:

Property 1 (Concavity and valid cycle times)**.** *Consider a port j, if $\sum_{i \in Fl(j)} \tilde{\alpha}^i$ is concave*[5] *and $T_j^{opt} > 0$, then $\mathcal{T}_j = \left[T_j^{opt}, +\infty\right)$ and $T_j^{opt} = T_j^{safe}$.*

5. $f : \mathbb{R}^+ \to \mathbb{R}^+$ is concave if $\forall x, y \in \mathbb{R}^+, \forall p \in [0,1], pf(x) + (1-p)f(y) \leq f(px + (1-p)y)$

The proof is given in Appendix F.

As mentioned in Section 2.4, it is common to simplify network calculus computations by replacing the exact arrival curves (which typically contain steps) by linear upper bounds. When incorporating clock non-idealities into the model, this leads to concave, piece-wise linear arrival curves as in (7). Applying this idea leads to a closed-form upper bound on T_j^{opt} and T_j^{safe}:

Definition/Property 3 (Local, Concave Cycle Time)**.** *Consider a port j, some burst and rate values b, r, and some clock imperfections characterized by Δ, ρ, and η such that $\sum_{i \in Fl(j)} \tilde{\alpha}^i(d) \leq \min\left(2r\Delta + b + rd, \eta + b + \rho rd\right)$, define*

$$T_j^{conc} \stackrel{def}{=} \min \begin{cases} \frac{b+2r\Delta+2R_jS+Bl_j}{R_j - r} \\ \\ \frac{b+\eta+2R_jS+Bl_j}{R_j - \rho r} \end{cases}, \tag{9}$$ {eq:Tlin}

- *then $T_j^{safe} \leq T_j^{conc}$.*
- *Moreover, if $\sum_{i \in Fl(j)} \tilde{\alpha}^i(d) = \min\left(2r\Delta + b + rd, \eta + b + \rho rd\right)$, then $T_j^{opt} = T_j^{safe} = T_j^{conc}$.*

The proof is by simple algebra for the first point. The second comes from the previous property, Property 1. It follows immediately that any $T \geq T_j^{\text{conc}}$ satisfies the local LEC condition at port j. We always have $T_j^{\text{opt}} \leq T_j^{\text{safe}} \leq T_j^{\text{conc}}$, but in general we have strict inequalities. In Section 4.3 we perform numerical experiments and see that the suboptimality of T_j^{conc} can be large. The effect of the clock imperfections is discussed in Section 4.4.

4.2 Global cycle time

Now, consider several CQF nodes in a network. A cycle T satisfies the *global* LEC condition if and only if it satisfies the local LEC condition for every output port j.

Definition/Property 4 (Global, Minimum Cycle Time)**.** *Define $\mathcal{T}$ as the set of values of the cycle T that satisfy* (8) *for all ports $j \in \mathcal{P}$,* i.e. *$\mathcal{T} = \bigcap_{j \in \mathcal{P}} \mathcal{T}_j$. Define $T^{opt} = \inf \mathcal{T}$. If $T^{opt} > 0$, then $T^{opt} \in \mathcal{T}$.*

Also, $T^{opt} \geq \max_{j \in \mathcal{P}} T_j^{opt}$ and the inequality is strict, in general.

The proof is in Appendix D.

Figure 10 shows an example where $T^{\text{opt}} > \max_{j \in \mathcal{P}} T_j^{\text{opt}}$. We consider two CQF ports of the network where $\alpha_1(d) = 2\left\lceil \frac{d}{2.5} \right\rceil$ is an arrival curve of the flows crossing the port 1, and $\alpha_2(d) = 3\left\lceil \frac{d}{5} \right\rceil$ is an arrival curve of the flows crossing the port 2. Locally, as illustrated, the optimal cycle times are $T_1^{\text{opt}} = 2\mu s$ and $T_2^{\text{opt}} = 3\mu s$. But, the optimal cycle time of the network is $T^{\text{opt}} = 4\mu s \neq \max(T_1^{\text{opt}}, T_2^{\text{opt}}) = 3\mu s$ ($T = 3\mu s$ is not a correct cycle time for the CQF port 1).

Again, it is still not margin-safe. For example, $T = 5.5\mu s$ is not a global correct cycle time because it is not a local correct cycle time for both ports.

Definition/Property 5 (Global, Margin-Safe Cycle Time)**.** *Consider a network composed of a set of CQF ports: $\mathcal{P}$. We say that T satisfies the global margin-safe condition if, and only if, for all $t \geq T$, $t \in \mathcal{T}$. Let $\mathcal{T}^{safe}$ denote the set of values that satisfy*

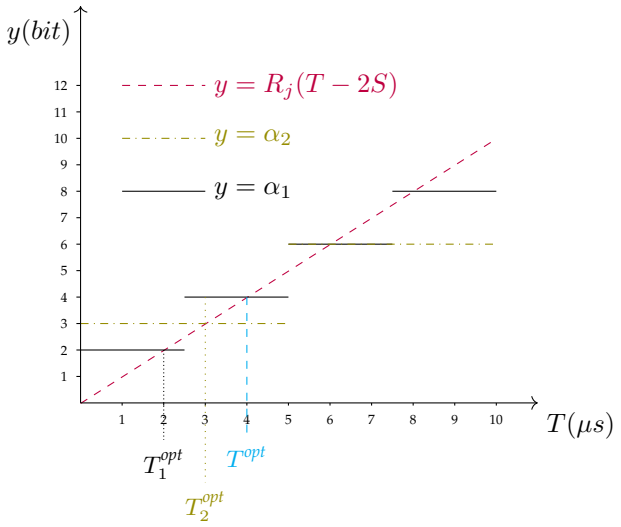


Fig. 10: Illustration of $\mathcal{T}_j$ for $j \in \{1,2\}$ with $\alpha_1(d) = 2\left\lceil \frac{d}{2.5} \right\rceil$, $\alpha_2(d) = 3\left\lceil \frac{d}{5} \right\rceil$, $R_j = 1bit/\mu s$, $S = 0$, $(\rho, \eta, \Delta) = (0,0,0)$, $\overline{L} = 0$.

the global margin-safe condition at port j and $T^{safe} = \inf \mathcal{T}^{safe}$. If $T^{safe} > 0$, then $\mathcal{T}^{safe} = \left[T^{safe}, +\infty\right)$. Furthermore

$$T^{safe} = \max_{j \in \mathcal{P}} T_j^{safe}. \tag{10}$$

The proof is in AppendixE.

Finally, assume that we overapproximate arrival curves by linear functions, and let T_j^{conc} be the resulting local, linear cycle time at port j. Then let $T^{\text{conc}} = \max_{j \in \mathcal{P}} T_j^{\text{conc}}$. It follows that T^{conc} is a margin-safe cycle time of the network and $T^{\text{opt}} \leq T^{\text{safe}} \leq T^{\text{conc}}$.

4.3 Numerical Experiments

We use some global parameters such that the link speed fixed at 100 Mb/s. Also, we consider that only CQF is present in our experiments *i.e.*, it doesn't exist any interference due to higher or lower priority flows ($\forall j, \overline{Bl}_j = 0$). Also, we consider a guard time equal to 10% of the cycle time T, according to the results from [11]. Finally, for the clock imperfections, we use the the standard values recalled in Section 2.3: $(\rho, \eta, \Delta) = (1.0001, 2ns, 1\mu s)$.

Then, we introduce a set of flow patterns as shown in Table 2. As we can see, each entry represents on average ca. one percent of the link load. We randomly add flows in our experiments.

To illustrate the problem of the choice of the cycle time, we consider 2 topologies : one node and a line of 16 nodes. The aim is to show the property of the different cycle times.

4.3.1 One node

First, we consider a simple CQF node. We compute the four cycle time ($T^{\text{opt}}, T^{\text{safe}}, T^{\text{conc}}$) and see the differences on this topology.

In this experiment, we start with one flow and then randomly add a flow from the set until we have 90 flows; at every step we compute the different cycle times (for 90 flows, the average load of the output link is 90%). Figure 11 illustrates the results of one configuration.

TABLE 2: Recapitulative table of the packet parameters

Size (byte)	Period (ms)	Load (% of R)
96	1	0.798
128	1	1.024
200	2	0.8
256	2	1.024
496	4	0.992
512	4	1.024
1000	8	1
1024	8	1.024
1504	12	1.003
1520	12	1.013

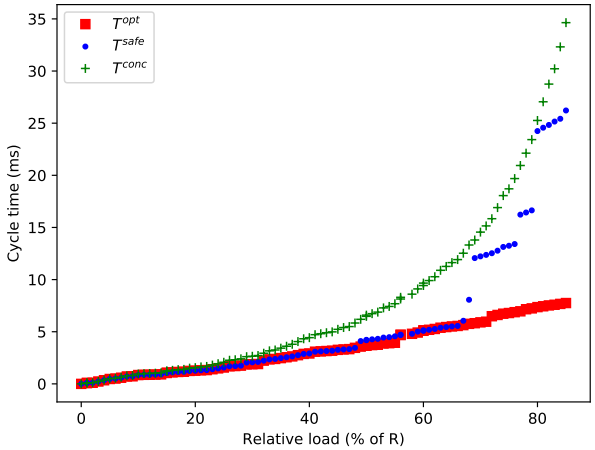


Fig. 11: Illustration of the latencies for one configuration of the OneNode topology.

To see the difference between the cycle times, we compare the ratios $\frac{T^{\text{safe}}}{T^{\text{opt}}}$ and $\frac{T^{\text{conc}}}{T^{\text{opt}}}$ in Figure 12.

Then, to see more relevant results, we made 100 configurations and for each one and for each number of flows, we compute the different cycle times. Then, we draw the boxplot of the latency values in Figure 13 and the ratio values in Figure 14 (the ratios $\frac{T^{\text{safe}}}{T^{\text{opt}}}$ are in blue and $\frac{T^{\text{conc}}}{T^{\text{opt}}}$ in green).

Interpretations: Using the two figures, Figure 13 and Figure 14, we can highlight some trends and remarks. The concave cycle time (here T^{conc}) starts to distance the others (more than 1.5 times the other cycle times) from 40 precent of the load whereas the difference between the margin-safe one (T^{safe}) and the optimal one (T^{opt}) is visible around 70 percent of the load. Even if the difference between the optimal safe one (T^{safe}) and the concave one (here, T^{conc}) seems to grow with the load, it is not a linear increase. Indeed, around 80 percent of the load, we can see that the difference can pass from on average 7ms to 1ms.

4.3.2 Line

Secondly, we want to see the difference between the cycle times in a Line of 16 nodes. Here, we will see if the

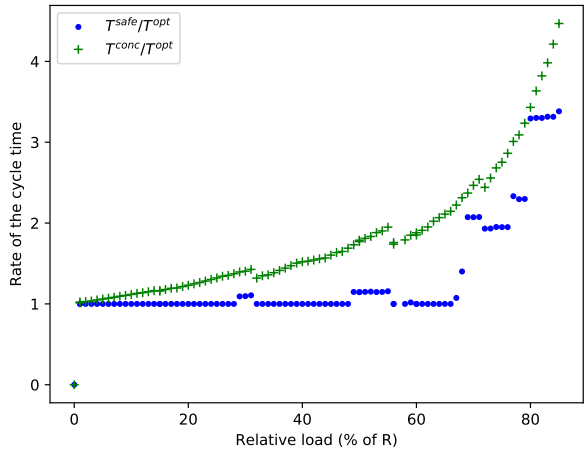


Fig. 12: Illustration of the ratios for one configuration of the OneNode topology ($\frac{T^{\text{safe}}}{T^{\text{opt}}}$ (in blue dots) and $\frac{T^{\text{conc}}}{T^{\text{opt}}}$ (in green crosses)).

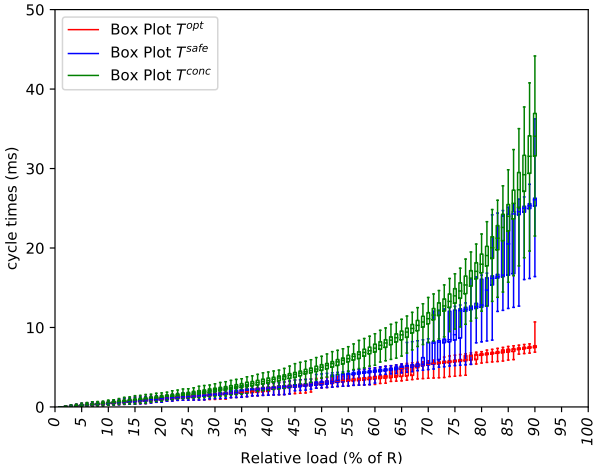


Fig. 13: Illustration of the latency results for 100 configurations of the OneNode topology.

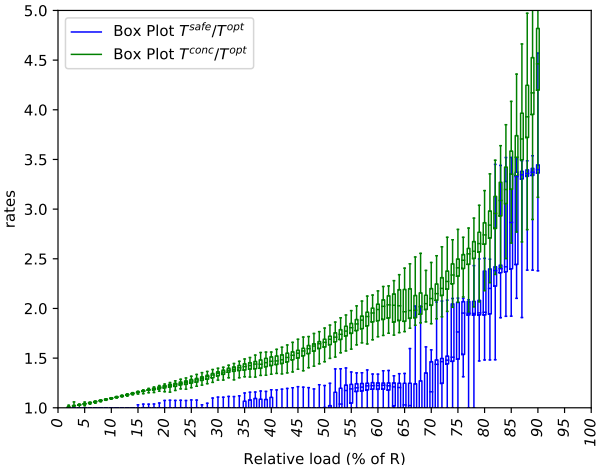


Fig. 14: Illustration of the ratio results for 100 configurations of the OneNode topology ($\frac{T^{\text{safe}}}{T^{\text{opt}}}$ (in blue) and $\frac{T^{\text{conc}}}{T^{\text{opt}}}$ (in green)).

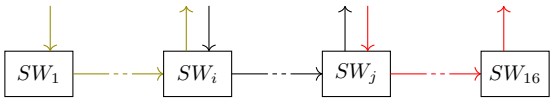


Fig. 15: Illustration of the line topology.

successive nodes influence the cycle times.

To this end, we randomly pick two integer (i, j) between 2 and 15 and three flows from the set defined in Table 2. We route each flow as follows (see Figure 15):

1) the first one starts at the first node (SW_1) and leaves the line at the switch i.
2) the second one enters in the line at the switch i (SW_i) and leaves the line at the switch j.
3) the third one enters in the line at the switch j (SW_j) and leaves the line at the last switch (SW_{16}).

The results for a single configuration of the line is illustrated in Figure 16.

To see the difference between the cycle times, we compare the ratio $\frac{T^{\text{safe}}}{T^{\text{opt}}}$ (in blue) and $\frac{T^{\text{conc}}}{T^{\text{opt}}}$ (in green) in Figure 17.

Then, to see more relevent results, we made 100 configurations and for each one and for each number of flows, we compute the different cycle times. Then, we draw the boxplot of the latency values in Figure 18 and the ratio values in Figure 19 (the ratios are $\frac{T^{\text{safe}}}{T^{\text{opt}}}$ (in blue) and $\frac{T^{\text{conc}}}{T^{\text{opt}}}$ (in green)).

Interpretations: The trends are similar to the experiments of the single node. In particular, we can see with these two figures, Figure 13 and Figure 14, that the concave cycle time (here, T^{conc}) starts to distance the others (more than 1.5 times the other cycle times) from the 50 precent of load and the difference between the margin-safe one and the optimal one is visible around 70 percent of load.

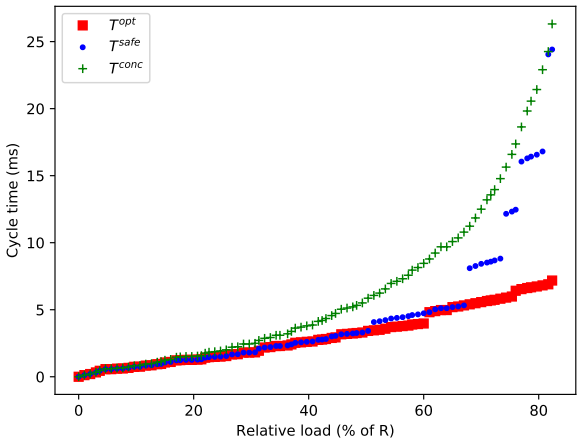


Fig. 16: Illustration of the latency results for one configuration of the Line of 16 nodes.

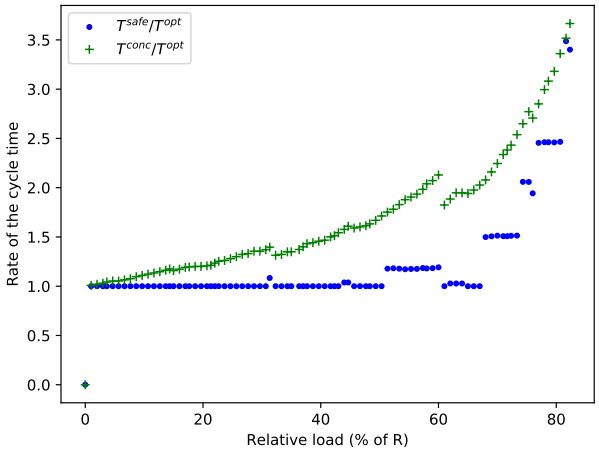


Fig. 17: Illustration of the ratio for one configuration of the Line topology ($\frac{T^{\text{safe}}}{T^{\text{opt}}}$ (in blue dots) and $\frac{T^{\text{conc}}}{T^{\text{opt}}}$ (in green crosses)).

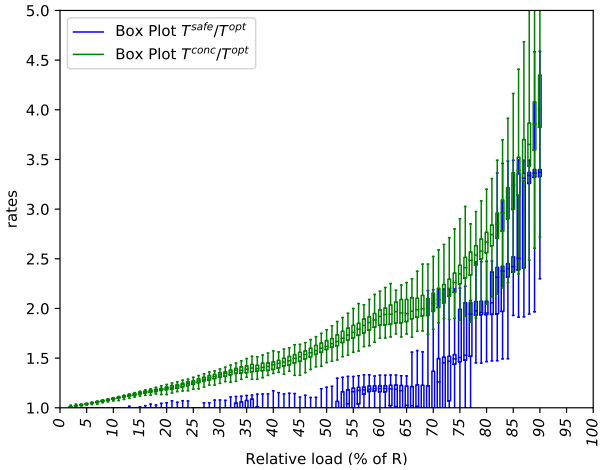


Fig. 19: Illustration of the ratio results for 100 configurations of the Line topology ($\frac{T^{\text{safe}}}{T^{\text{opt}}}$ (in blue) and $\frac{T^{\text{conc}}}{T^{\text{opt}}}$ (in green)).

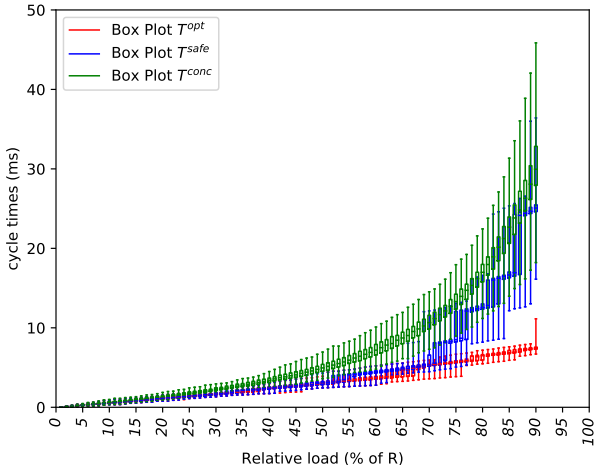


Fig. 18: Illustration of the latency results for 100 configurations of the Line topology.

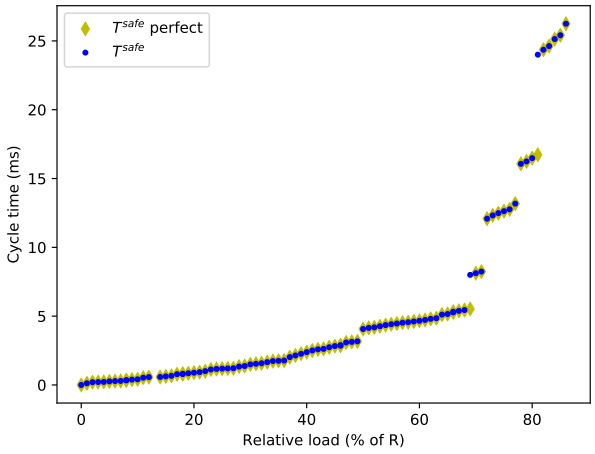


Fig. 20: Illustration of the latencies for one configuration of the OneNode topology with and without clock imperfections.

4.4 Influence of clock imperfections

In this section, we numerically evaluate the influence of the clock imperfections by comparing our results with those that would be obtained if we would assume clocks to be perfect. The results with the latter assumption are obtained to setting $\rho = 1$ and $\eta = 0$. Specifically, we draw, using the same configuration as in Figure 11, on the same graph, the computed margin-safe cycle time with and without considering the clock imperfections (T^{safe} and T^{safe} perfect respectively) as shown in Figure 20. We can see, for this configuration sample, the effect of clock imperfection is negligible except for two points, where it is large: for one point (at around 70% load) the values are 5.525 ms (perfect clock) and 8 ms (with clock imperfection); for the other (at around 80% load) the values are 16.727 ms (perfect clock) and 23.998 ms (with clock imperfection).

As before, we repeated the experiment and drew 100 configuration samples (i.e, with 8900 cycle times in total, with 89 cycle times per configuration). We found 22 points (out of 8900) where the margin-safe cycle times which are different if we consider or not the clock imperfections. We also computed the minimum cycle times and found only two points where they differ. However, the difference at these 24 point is not negligible, ranging from $7\mu s$ to $7271\mu s$ with an average of $642\mu s$.

Regarding the concave cycle times, all the computed cycle times are different, but with a small difference, in average equal to $4.37\mu s$.

5 Related work

The principle of cutting the time in equals slices, and to forward a packet in the slice following its reception slice has been first presented in [1]. It was designed to reduced the jitter and buffer occupancy w.r.t. to FIFO policy. Note that

the delay bound in this first work was $[hT, 2hT]$, whereas the modern evaluation is $[(h-1)t, (h+1)T]$. The extension to multiple cycle has been proposed in [26].

CQF can be used to transfer two kinds of flows: either time-triggered flows (a.k.a. scheduled flows), whose emission time of each packet is statically known in advance or for asynchronous flows (a.k.a. sporadic flows) where only admissible burst and rate are known [5, § 1.2].

The initial proposal in [26] was targeting asynchronous flows, but multiple works have been done on the building of a global schedule adapted to CQF. The Injection Time Planning (ITP) mechanism computes the global cycle time and per-flow offsets for periodic flows in order to maximize the admissible load while satisfying this constraint [3]. The approach is generalized in [4] by adding in each End-System an "adapter" in charge of implementing the injection using a list of queues, and also by providing an online algorithms. It assumes that all offsets are equals, and the guard band only has to absorb the clock synchronization precision.

Several extensions of CQF have been proposed, like CQF 3-queues [27], Paternoster [28], Scalable Deterministic Forwarding (SDF, [29]) and Cycle Specified Queuing and Forwarding (CSQF, [30]). A global survey can be found in [6].

The performances of CQF are compared with Time-Aware Shaper (TAS), Multi-CQF (CQF with multiple cycle, [5], not addressed in this paper) and CSQF in [31]. It shows the benefits of these extensions w.r.t. the simple CQF. Strictly periodic flows are considered, but with unknown offsets. The problem of finding an adequate cycle time is discussed but its configuration is postponed to further works [31, § V.C].

Also note that all these extensions use 3 or even 4 queues, whereas Ethernet/TSN offers only 8 queues (or even less, depending on the implementation). Since a real-time network may have many types of traffic, each with specific requirements, the mapping from traffic types to only 8 TSN classes may become an issue [32].

A comparison between CQF and CBS on an automotive use case is done by simulation in [33], assuming strictly periodic flows.

6 Conclusion

This study focuses on the use of CQF for carrying asynchronous flows.

One major trade-off of CQF is the choice of cycle time: a short cycle time reduces the latency and jitter but can only admit small bursts, limiting the number of admissible flows to a small part of the bandwidth.

This paper proves (in Theorem 1) that the burstiness does not increase along the path, even in the presence of nonideal clocks, allowing one to check the cycle time by only considering the flow characteristics at network input.

It also points out that the optimal cycle time (T^{opt}) is not margin-safe (*i.e.* is not robust to small changes: for $\varepsilon > 0$, $T^{\text{opt}} + \varepsilon$ is not necessarily a valid cycle time), which is a serious industrial limitation, since it complicates the possible evolutions of a system.

We also provide a simple expression (T^{conc}) for an upper-bound on the margin-safe cycle time and provide several evaluations of its suboptimality. These experiments show that the differences between the optimal cycle time (T^{opt}) and the best margin-safe cycle time (T^{safe}) grow with the network load, but the difference between T^{safe} and T^{conc} is not monotonous with the network load.

Finally, regarding the clock imperfections, we highlight that they often do not influence the cycle times. However, when they do, the difference between the ones that ignores the clock imperfections and the true ones can be large. This suggests that the clock imperfections should not be overlooked.

References

[1] S. Golestani, "Congestion-free transmission of real-time traffic in packet networks," in *Proc. of the Ninth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '90)*, 1990, pp. 527–536 vol.2.

[2] "IEEE Standard for Local and metropolitan area networks–Bridges and Bridged Networks–Amendment 29: Cyclic Queuing and Forwarding," IEEE, IEEE Standard 802.1Qch, 2017.

[3] J. Yan, W. Quan, X. Jiang, and Z. Sun, "Injection time planning: Making cqf practical in time-sensitive networking," in *Proc. of the 39th IEEE Conference on Computer Communications (INFOCOM 2020)*, 2020, pp. 616–625.

[4] W. Quan, J. Yan, X. Jiang, and Z. Sun, "On-line traffic scheduling optimization in ieee 802.1Qch based time-sensitive networks," in *Proc. of the Int. Conf. on High Performance Computing and Communications; Int. Conf. on Smart City; 6th Int. Conf. on Data Science and Systems (HPCC/SmartCity/DSS)*. IEEE, 2020, pp. 369–376.

[5] N. Finn, "Mutiple cyclic queuing and forwarding," IEEE 802.1 Working Group, Tech. Rep. df-finn-multiple-CQF-0919-v01, September 2019. [Online]. Available: https://www.ieee802.org/1/files/public/docs2019/df-finn-multiple-CQF-0919-v01.pdf

[6] A. Nasrallah, V. Balasubramanian, A. S. Thyagaturu, M. Reisslein, and H. ElBakoury, "Cyclic queuing and forwarding for large scale deterministic networks: A survey," *ArXiv, vol. abs/1905.08478*, 2019.

[7] B. Jonsson, S. Perathoner, L. Thiele, and W. Yi, "Cyclic dependencies in modular performance analysis," in *Proc. of the 8th Int. Conf on Embedded Software (EMSOFT'08)*. ACM Press, 2008, pp. 179–188. [Online]. Available: http://doi.acm.org/10.1145/1450058.1450083

[8] D. Starobinski, M. Karpovsky, , and L. Zakrevski, "Application of network calculus to general topologies using turn-prohibition," in *Proc. of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies*, June 2002, pp. 411–421.

[9] L. Thomas, J.-Y. Le Boudec, and A. Mifdaoui, "On cyclic dependencies and regulators in time-sensitive networks," in *IEEE Real-Time Systems Symposium, RTSS 2019, Hong Kong, SAR, China, December 3-6, 2019*. IEEE, 2019, pp. 299–311. [Online]. Available: https://doi.org/10.1109/RTSS46320.2019.00035

[10] S. Plassart and J.-Y. L. Boudec, "Equivalent versions of total flow analysis," 2021. [Online]. Available: https://arxiv.org/abs/2111.01827

[11] D. Guidolin--Pina, M. Boyer, and J.-Y. Le Boudec, "Configuration of guard band and offsets in cyclic queuing and forwarding," Tech. Rep., Sep. 2022, forthcoming. [Online]. Available: https://hal.archives-ouvertes.fr/hal-03772877

[12] A. Nasrallah, V. Balasubramanian, A. S. Thyagaturu, M. Reisslein, and H. ElBakoury, "Cyclic queuing and forwarding for large scale deterministic networks: A survey," *ArXiv, vol. abs/1905.08478*, 2019.

[13] "IEEE standard for local and metropolitan area networks – bridges and bridged networks," IEEE, IEEE Standard 802.1Q, 2018.

[14] "IEEE standard for local and metropolitan area networks–bridges and bridged networks–amendment 28: Per-stream filtering and policing," IEEE, Standard 802.1Qci, 2017.

[15] "IEEE standard for local and metropolitan area networks – bridges and bridged networks – amendment 26: Frame preemption," IEEE Standard 802.1Qbu, 2016.

[16] IEEE, "IEEE standard for ethernet amendment 5: Specification and management parameters for interspersing express traffic," IEEE Standard 802.3br, 2016.

[17] D. Thiele and R. Ernst, "Formal worst-case performance analysis of time-sensitive ethernet with frame preemption," in *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2016, pp. 1–9.

[18] S. S. Craciunas, R. S. Oliver, M. Chmelík, and W. Steiner, "Scheduling real-time communication in IEEE 802.1Qbv time sensitive networks," in *Proceedings of the 24th International Conference on Real-Time Networks and Systems (RTNS'16)*, ser. RTNS'16. New York, NY, USA: Association for Computing Machinery, 2016, pp. 183–192. [Online]. Available: https://doi.org/10.1145/2997465.2997470

[19] P.-J. Chaine and M. Boyer, "Shortening gate closing time to limit bandwidth waste when implementing time-triggered scheduling in tas/tsn," in *Proc. of the 15th Junior Researcher Workshop on Real-Time Computing*, Paris, France, 2022. [Online]. Available: https://rtns2022.inria.fr/files/2022/06/proceedings_jrwrtc2022_final.pdf

[20] Y. Huang, S. Wang, B. Wu, T. Huang, and Y. Liu, "TACQ: enabling zero-jitter for cyclic-queuing and forwarding in time-sensitive networks," in *ICC 2021 - IEEE International Conference on Communications*, 2021, pp. 1–6.

[21] M. Boyer and P. Roux, "Embedding network calculus and event stream theory in a common model," in *Proc. of the 21st IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA 2016)*, Berlin, Germany, September 2016.

[22] "IEEE/ISO/IEC International Standard for Information technology–Telecommunications and information exchange between systems–Local and metropolitan area networks–Part 1AS:Timing and synchronization for time-sensitive applications in bridged local area networks, ISO/IEC/IEEE 8802-1AS:2021(E)," 2021.

[23] L. Thomas and J.-Y. Le Boudec, "On time synchronization issues in time-sensitive networks with regulators and nonideal clocks," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 4, no. 2, pp. 1–41, 2020.

[24] S. M. Tabatabaee, M. Boyer, J.-Y. Le Boudec, and J. Migge, "Efficient and accurate handling of periodic flows in time-sensitive networks," in *2023 IEEE 29th Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 2023, pp. 303–315.

[25] A. Bouillard, M. Boyer, and E. Le Corronc, *Deterministic network calculus: From theory to practical implementation*. John Wiley & Sons, 2018.

[26] S. J. Golestani, "A stop-and-go queueing framework for congestion management," in *Proceedings of the ACM symposium on Communications architectures & protocols*, 1990, pp. 8–18.

[27] N. Finn, J.-Y. Le Boudec, E. Mohammadpour, J. Zhang, J. Farkas, and B. Varga, "Detnet bounded latency-02," https://www.ietf.org/proceedings/103/slides/slides-103-detnet-07-detnet-bounded-latency-02.pdf, Bangkok, November 8th 2018.

[28] M. Seaman, "Paternoster policing and scheduling," IEEE, Tech. Rep., May 2019, revision 2.1. [Online]. Available: https://grouper.ieee.org/groups/802/1/files/public/docs2019/cr-seaman-paternoster-policing-scheduling-0519-v04.pdf

[29] L. Qiang, X. Geng, B. Liu, T. Eckert, L. Geng, and G. Li, "Large-scale deterministic ip network," IETF, Internet-Draft draft-qiang-detnet-large-scale-detnet-05, 2019. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-qiang-detnet-large-scale-detnet-05

[30] M. G. Chen, G. Xuesong, and Z. Li, "Segment routing (sr) based bounded latency," IETF, Internet-Draft draft-chen-detnet-sr-based-bounded-latency-01, May 2019. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-chen-detnet-sr-based-bounded-latency-01

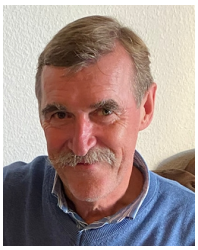
[31] K. Alexandris, P. Pop, and T. Wang, "Configuration and evaluation of multi-cqf shapers in IEEE 802.1 time-sensitive networking (tsn)," *IEEE Access*, 2022.

[32] "Time-sensitive networking profile for industrial automation," IEEE 802, Tech. Rep., 2021.

[33] L. Leonardi, L. L. Bello, and G. Patti, "Performance assessment of the IEEE 802.1Qch in an automotive scenario," in *Proc. of the AEIT Int. Conf. of Electrical and Electronic Technologies for Automotive (AEIT AUTOMOTIVE)*, 2020, pp. 1–6.

Damien GUIDOLIN--PINA is an industrial PhD student at RealTime-at-Work since 2021, with academic enrollment at ISAE SUPAERO. His academic supervisor is Marc Boyer (Onera/DTIS). He received an engineering degree from ISAE ENSMA, Poitiers, France, in 2020. He then joined RealTime-at-Work, a French software editor helping OEMs and Tier1s design safe & cost optimized Electrical/Electronic (E/E) communication architectures for 10 months as a software engineer working on implementing some IEEE TSN features in RTaW-PEGASE, an industry-leading platform for the design, configuration, and simulation of embedded communication architectures. His research focuses on the performance of real-time communication systems, mainly using the Network Calculus theory.

Marc BOYER received the Engineering degree in computer science from Toulouse INP-ENSEEIHT, Toulouse, France, in 1996, and the Ph.D. degree in computer science from the Université Paul Sabatier Toulouse III, Toulouse, in 2001. He was an Assistant Professor with the Network Department, Toulouse INP-ENSEEIHT, from 2002 to 2008. He is currently a Research Scientist with the Office National d'Etudes et de Recherches Aérospatiales (ONERA), Toulouse. His research interests focuses on embedded networks (AFDX, TSN) and the performances of these networks, mainly using the network calculus theory.



Jean-Yves LE BOUDEC is honorary professor at EPFL and fellow of the IEEE. He graduated from Ecole Normale Supérieure de Saint-Cloud, Paris, where he obtained the Agrégation in Mathematics with rank 4 in 1980, and received his doctorate in 1984 from the University of Rennes, France. From 1984 to 1987 he was with INSA/IRISA, Rennes. In 1987 he joined Bell Northern Research, Ottawa, Canada, as a member of scientific staff in the Network and Product Traffic Design Department. In 1988, he joined the IBM Zurich Research Laboratory where he was manager of the Customer Premises Network Department. In 1994 he became associate professor at EPFL. His interests are in the performance and architecture of communication systems and smart grids. He co-authored a book on network calculus, which serves as a foundation for deterministic networking, an introductory textbook on Information Sciences, and is the author of the book "Performance Evaluation". He received numerous awards, among which the IEEE millenium medal, the Infocom Best Paper award, the ACM Sigmetrics Best Paper award, the ACM Conext Best Paper Award, the IEEE Communication Society William R. Bennett Prize, the IEEE Security and Privacy Test-of-Time award and the EPFL I&C Best Teacher Award.

APPENDIX A
PROOF OF THEOREM 1

To prove this theorem, we need to introduce first further notations. In addition, we have to bound the cycle gap between two nodes. We also introduce the per. Finally, we prove the main theorem.

A.1 Further notations

A.1.1 Notation for Routing

The routing of flows is static and is described by means of a function pred(), such that $\text{pred}(i,j) = j'$ means that flow i goes from output port j' to output port j. For example, on Fig. 4, and for a flow f (resp. g) going from ES1 (resp. ES1) to ES3 trough SW, and using output ports p_{ES1} (resp. p_{ES2}) and p_{SW}, it comes $\text{pred}(f, p_{SW}) = p_{ES1}$ and $\text{pred}(f, p_{ES1}) = \perp$ where $\perp$ means f has no predecessor passing through p_{ES1}.

A.1.2 Notation for Backlog

As mentionned in the sketch of proof, we also introduce $B_{j,0}(k), B_{j,1}(k)$: the backlog in $q_{j,0}, q_{j,1}$ respectively at the end of cycle $k \geq 0$, with the convention that $\forall j : B_{j,0}(-1) = B_{j,0}(-2) = B_{j,1}(-1) = B_{j,1}(-2) = 0$. To find the expression of the backlog, we just translate the presentation of the CQF per port behaviour of Section 2.1 into mathematical expressions.

This backlog counts not only the packets themselves, but also the overhead of the physical layer[6].

Regarding a even cycle, the backlog in the queues at a cycle k depends on the backlog of the previous one, the cycle $k-1$. Also, acccording to Section 2, the backlog in the queue q_0 depends on what is leaving from the queue during this cycle k, *i.e.*

$$B_{j,0}(k) = B_{j,0}(k-1) - \sum_{i \in Fl(j)} \text{Out}_{i,j}(k),$$

and the backlog in the queue q_1 depends on what is entering in the queue, *i.e.*

$$B_{j,1}(k) = B_{j,1}(k-1) + \sum_{i \in Fl(j)} \text{In}_{i,j}(k).$$

Reciprocally, for the odd queue, we have

$$B_{j,0}(k) = B_{j,0}(k-1) + \sum_{i \in Fl(j)} \text{In}_{i,j}(k),$$
$$B_{j,1}(k) = B_{j,1}(k-1) - \sum_{i \in Fl(j)} \text{Out}_{i,j}(k).$$

Thus, using the modulo notation, we have

$$B_{j,(k+1)\,\%\,2}(k) = B_{j,(k+1)\,\%\,2}(k-1) + \sum_{i \in Fl(j)} \text{In}_{i,j}(k), \tag{11}$$

{eq:B:in}

$$B_{j,k\,\%\,2}(k) = B_{j,k\,\%\,2}(k-1) - \sum_{i \in Fl(j)} \text{Out}_{i,j}(k). \tag{12}$$

{eq:B:out}

6. In the switch memory, only the Ethernet packet is stored, from MAC destination address up to the CRC/FCS. But when accounting for the time usage of the physical layer, one have to count also the preamble and the inter packet gap.

A.2 Cycle gap

Formally, we assume that for any port j, and any output port j' in the next switch, for any cycle index k at j, there exists an integer δ such that all packets sent by j in cycle k are received by node j' in cycle $k+\delta$ (this property is proved in [11, §4]). And since this topology relations is encoded in the pred relation, it becomes:

$$\forall j \in \mathcal{P}, \forall i \in Fl(j), \exists \delta_{\text{pred}(i,j),j} \in \mathbb{N}, \forall k \in \mathbb{N}, \quad \text{Out}^i_{\text{pred}(i,j)}(k - \delta_{\text{pred}(i,j),j}) = \text{In}^i_j(k). \tag{13}$$

{eq:out-is-ne

We bound this δ between two nodes.

Lemma 1. *Consider a system with all nodes time aligned. Let $j \in \mathcal{P}$ be a port, $i \in Fl(j)$ be a flow and $\delta_{pred(i,j),j} \in \mathbb{N}$ define as in Equation* (13)*. Then, $\delta_{pred(i,j),j} \geq -1$.*

Moreover, it exists configurations where $\delta_{pred(i,j),j} = -1$.

Proof of $\delta_{...} \geq -1$. Consider a system with all nodes time aligned. Let $j \in \mathcal{P}$ be a port, $i \in Fl(j)$ a flow and $\delta_{\text{pred}(i,j),j} \in \mathbb{N}$ defining as in Equation (13).

$\delta_{\text{pred}(i,j),j}$ exists for any valid clock trajectory and the perfect clock is a valid trajectory ($\Delta = 0, \rho = 1, \eta = 0$). Then, from the necessary theorem in [11, §4.2],

$$\delta_{\text{pred}(i,j),j} = \left\lfloor \frac{T - S + \overline{P}_{i,\text{pred}(i,j)} + \overline{z}_j + o_i - o_j}{T} \right\rfloor \tag{14}$$

where $\overline{P}_{i,\text{pred}(i,j)}, \underline{P}_{i,\text{pred}(i,j)}$ is the upper and lower bound on propagation time, $\overline{z}_j, \underline{z}_j$ is the upper and lower bound on switching time and $\underline{E}_{i,\text{pred}(i,j)}$ is the minimum transmission time of CQF packet from $\text{pred}(i,j)$ to j.

By construction, we have $T - S + \overline{P}_{i,\text{pred}(i,j)} + \overline{z}_j + o_i - o_j \geq \underline{E}_{i,\text{pred}(i,j)} + S + \underline{P}_{i,\text{pred}(i,j)} + \underline{z}_j + o_i - o_j$. Consequently, to find a lower bound, we will use the greater one. Also, according to Section 2.4, $0 \leq o < T$ then $-T < o_i - o_j < T$ and according to [11, §3], $0 \leq S \leq \frac{T - \overline{E}}{2}$. Consequently,

$$\delta_{\text{pred}(i,j),j} = \left\lfloor \frac{T - S + \overline{P}_{i,\text{pred}(i,j)} + \overline{z}_j + o_i - o_j}{T} \right\rfloor \tag{15}$$

$$\Leftrightarrow \frac{T - S + \overline{P}_{i,\text{pred}(i,j)} + \overline{z}_j + o_i - o_j}{T} - 1 < \delta_{\text{pred}(i,j),j} \leq \frac{T - S + \overline{P}_{i,\text{pred}(i,j)} + \overline{z}_j + o_i - o_j}{T} \tag{16}$$

$$\implies \frac{T - S + \overline{P}_{i,\text{pred}(i,j)} + \overline{z}_j + o_i - o_j}{T} - 1 < \delta_{\text{pred}(i,j),j} \tag{17}$$

$$\implies \frac{T + \overline{P}_{i,\text{pred}(i,j)} + \overline{z}_j - T}{T} - 2 < \delta_{\text{pred}(i,j),j} \tag{18}$$

$$\implies \frac{\overline{P}_{i,\text{pred}(i,j)} + \overline{z}_j}{T} - 2 < \delta_{\text{pred}(i,j),j} \tag{19}$$

$$\implies \delta_{\text{pred}(i,j),j} > -2 \tag{20}$$

But $\delta_{\text{pred}(i,j),j} \in \mathbb{N}$ then $\delta_{\text{pred}(i,j),j} \geq -1$. □

Proof that $\delta_{...} = -1$ is a possible value. Let construct an example where $\delta_{\text{pred}(i,j),j} = -1$. Consider an ideal configuration with perfect clocks, no switching time and no jitter on the

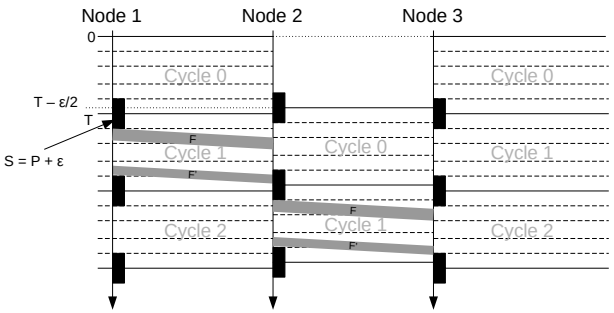


Fig. 21: Illustration of a configuration where $\delta_{i,\text{pred}i,j} = -1$.

propagation. Then, according to the necessary theorem in [11, §4.2],

$$\delta_{\text{pred}(i,j),j} = \left\lfloor \frac{T - S + P_{i,\text{pred}(i,j)} + o_i - o_j}{T} \right\rfloor = -1 \quad (21)$$

$$\Rightarrow -1 \leq \frac{T - S + P_{i,\text{pred}(i,j)} + o_i - o_j}{T} < 0 \quad (22)$$

$$\Leftrightarrow -T \leq T - S + P_{i,\text{pred}(i,j)} + o_i - o_j < 0 \quad (23)$$

Such case can appear, for example, in a configuration with 3 nodes in sequence: Node 1, Node 2, Node 3. Let $\varepsilon = 0,01.P_{i,\text{pred}(i,j)}$, let offsets $o_1 = 0$, $o_1 = T - \frac{\varepsilon}{2}$ and $o_3 = 0$, and $S = P_{i,\text{pred}(i,j)} + \varepsilon$. Then,

$$T - S + P_{i,\text{pred}(i,j)} + o_i - o_j = -\frac{\varepsilon}{2}. \quad (24)$$

Figure 21 illustrates this example. The packet F can't be sent before the openning guard band, here $T + S = T + P - \varepsilon$ and the last packet F' is sent just before the closing guard band, here $2T - S = 2T - P - \varepsilon$. Then, the last bit of the last packet arrives at $2T - P - \varepsilon + P = 2T - \varepsilon$ and the cycle of Node 2 ends at time $2T - \frac{\varepsilon}{2}$, just after the packet F' arrives. The same way, Node 2 sends packets to Node 3.

Consequently, Node 2 receives packets from a greater cycle from Node 1, *i.e.* $\delta_{1,2} = -1$ and Node 3 receives packets from a lower cycle from Node 2 *i.e.* $\delta_{1,2} = 1$.

□

A.3 Local LEC Condition

The next property represents the core of CQF: if the queue $(k+1)\,\%\,2$ is empty at the end of cycle $k-1$ and if the sum of incoming amount of CQF data is "not too large" at cycle k w.r.t. cycle time and blocking (cf. Section 2.2), then the output in cycle $k+1$ is equal to the input at cycle k and the queue $(k+1)\,\%\,2$ is again empty at end of cycle $k+1$.

Property 2 (Per cycle LEC condition)**.** *Let $j \in \mathcal{P}$ be a port and $k \in \mathbb{N}$ be a cycle. If $B_{j,(k+1)\,\%\,2}(k-1) = 0$ and*

$$\sum_{i \in Fl(j)} In_j^i(k) \leq R_j(T - 2S) - Bl_{j,k} \quad (25)$$

then, $B_{j,k\,\%\,2}(k) = 0$ and

$$\forall i \in \mathcal{F}, Out_j^i(k+1) = In_j^i(k). \quad (26)$$

Proof. Let j a port of $\mathcal{P}$ and k a cycle. Since the proof involves only the port j, we omit the j exponent and subscripts for clock c, backlog B and throughput R.

The first part consist in evaluating the number of bits that could be send by the port during one cycle. The common notion of throughput given in "bit per second" relies on the notion of second, but the implementation relies on local clocks, and the bit time is based on the local clock (the IFG and preamble being in charge of absorbing a clock drift between the emitting and receiving nodes). Then, we consider that, in a cycle $k+1$, *i.e.* between the instants when the local clock value is $(k+1)T+S+o$ and $(k+2)T-S+o$, the port can emit $R(T-2S)$ bits.

When local clock value is $(k+1)T + S + o$, when the cycle $k+1$ starts, the gate of the queue $q_{(k+1)\,\%\,2}$ opens. The backlog of $q_{(k+1)\,\%\,2}$ at end of cycle k is $B_{j,(k+1)\,\%\,2}(k) = B_{j,(k+1)\,\%\,2}(k-1) + \sum_{i \in Fl(j)} In_j^i(k)$ (from eq. 11). By hypothesis, $B_{j,(k+1)\,\%\,2}(k-1) = 0$, so $B_{j,(k+1)\,\%\,2}(k) = \sum_{i \in Fl(j)} In_j^i(k)$.

Since the sum of the CQF backlog ($\sum_{i \in Fl(j)} In_j^i(k)$) and the blocking ($Bl_k$) is less than the number of bit opportunities in the window $R(T-2S)$ bits, it may appear obvious than all the backlog can be transmitted. But the blocking factor Bl_k does not account for next-closing-event blocking. What we have to prove is that it does not occur in our case.

The key argument is that, since all CQF packets to be sent in this window are in the queue from the start of the window, there is no idle time up to the transmission of the last CQF packet (all bit transmission opportunities are either used by CQF or counted in the blocking term Bl_k) and there always is enough time for each CQF packet to be fully transmitted before the closing event of the window.

Formally, assume they are n packets in queue $q_{(k+1)\,\%\,2}$ at time $t_0 = (k+1)T + S + o$, the i-th packet being of size L_i (including the preamble and inter packet gap overhead), starting with $i = 1$. Then $\sum_{i=1}^{n} L_i = B_{j,(k+1)\,\%\,2}(k) = \sum_{i \in Fl(j)} In_j^i(k)$. Let t_i be the start of transmission of the i-th CQF packet (without any assumption at this step that all the n packets will be transmitted) and t_{n+1} the closing time of the window. Let bl_i being the amount of blocking between t_i and t_{i+1} (if any) or t_{n+1}.

By induction, let show that for all $i \in [1, n]$, the i-th packet can be send, and use the property that $t_i = t_0 + \frac{\sum_{m=1}^{i} bl_m + \sum_{m=1}^{i-1} L_m}{R}$. For $i = 1$, consider the system at t_0. The first packet is ready to be sent. It may experience some blocking, bl_0, but $bl_0 \leq Bl_k$, so, at time $t_0 + \frac{bl_0}{R}$, the remaining time is $t_{n+1} - (t_0 + bl_0) = (2T - S) - \frac{bl_0}{R} \leq (2T - S) - \frac{Bl_k}{R}$ which is not less than $\frac{\sum_{i \in Fl(j)} In_j^i(k)}{R}$. Then, it remains enough transmission bit opportunity up to the end of the window and packet L_1 can be sent.

For any $p > 1$, at time $t_{p-1} + \frac{bl_p}{R} = \frac{\sum_{m=1}^{p} bl_m + \sum_{m=1}^{p-1} L_m}{R}$, the remaining time is

$$t_{n+1} - (t_p + bl_p) = (2T - S) - \frac{\sum_{m=1}^{p} bl_m + \sum_{m=1}^{p-1} L_m}{R} \quad (27)$$

$$\leq (2T - S) - \frac{Bl_k + \sum_{m=1}^{p-1} L_m}{R} \quad (28)$$

which is not less than $\frac{\sum_{i \in Fl(j)} In_j^i(k) - \sum_{m=1}^{p-1} L_m}{R} = \frac{\sum_{m=p}^{n} L_m}{R}$. Then, it remains enough transmission bit opportunity up to the end of the window and packet L_p can be sent.

□

A.4 Induction property

First, let $l(i,j)$ be the lenght of the path from the source of the flow i until the port j. By convention, we fix $l(i,j)=0$ if the port j is not the path of the flow i.

Hence, we have the following relation: $l(i,j) = l(i,\text{pred}(i,j))+1$.

Then, we introduce the property $P(k,d)$ defined as

$$\forall j \in \mathcal{P}, \forall i \in \text{Fl}(j),\ |\ l(i,j) \leq d, \tag{29}$$

{eq:P1} $$\text{In}_j^i(k) \leq \tilde{\alpha}^i(T) \tag{P1}$$

{eq:P2} $$\&\ B_{j,(k+1)\,\%\,2}(k-1) = 0 \tag{P2}$$

{eq:P3} $$\&\ B_{j,(k+1)\,\%\,2}(k) \leq R_j(T-2S) - \overline{Bl}_j \tag{P3}$$

{eq:P4} $$\&\ \text{In}_j^i(k) = \text{Out}_j^i(k+1). \tag{P4}$$

Let us prove this property by induction. To do that, we introduce a lexicographical order defined as

$$(k,d) > (k',d') \Leftrightarrow \begin{cases} k > k' \\ k = k'\ \&\ d > d' \end{cases}. \tag{30}$$

- For $k=0$ and any $d \in \mathbb{N}$, let $j \in \mathcal{P}$ be a port and $i \in \text{Fl}(j)$ a flow such that $l(i,j) \leq d$.
  - P1:
    - If the flow i comes from a end-system ($\text{pred}(i,j) = \perp$, $j = \text{fst}(i)$) then $\text{In}_j^i(0)$ is the amount of data send by the flow i on the first interval. By definition of the arrival curve enlarged due the clock nonidealities (Equation 6), $\text{In}_j^i(0) \leq \tilde{\alpha}^i(T)$.
    - Else, the flow i comes from a bridge. However, all the bridge queues of the network are empty at the first cycle. So $\text{Out}^i_{\text{pred}(i,j)} = 0$ and $\text{In}_j^i(k) = 0 \leq \tilde{\alpha}^i(T)$.

    Thus,
    $$\forall j \in \mathcal{P}, \forall i \in \text{Fl}(j), \text{In}_j^i(0) \leq \tilde{\alpha}^i(T) \tag{31}$$
  - P2: By definition of B, $B_{j,1}(-1) = 0$.
  - P3: By definition of B,
    $$B_{j,1}(0) \overset{(12)}{=} B_{j,1}(-1) + \sum_{i \in Fl(j)} \text{In}_j^i(0) \tag{32}$$
    $$\overset{(P2)}{=} \sum_{i \in Fl(j)} \text{In}_j^i(0) \tag{33}$$
    $$\overset{(P1)}{\leq} \sum_{i \in Fl(j)} \tilde{\alpha}^i(T) \tag{34}$$
    $$\overset{(8)}{\leq} R_j(T-2S) - \overline{Bl}_j. \tag{35}$$
  - P4: From eq. (8), (P1) and (1) it directly comes
    $$\sum_{i \in Fl(j)} In_j^i(0) \leq R_j(T-2S) - Bl_{j,k}, \tag{36}$$
    that, using Prop. 2 knowing that $B_{j,1}(-1) = 0$, implies $\forall j \in \mathcal{P}, \forall i \in \text{Fl}(j) : \text{In}_j^i(0) = \text{Out}_j^i(1)$.
    Then, $\forall d \in \mathbb{N}, P(0,d)$ is true.
- Induction step: let $k > 0, d > 0$, assume $\forall (k',d') < (k,d), P(k',d')$. Let $j \in \mathcal{P}$ be a port and $i \in \text{Fl}(j)$ be a flow such that $l(i,j) \leq d$.
  - P1:
    - If the flow i comes from an end-system, like in the case $k=0$, $\text{In}_j^i(k+1) \leq \tilde{\alpha}^i(T)$.
    - If the flow i comes from a switch, since the cycles are time aligned, cf. eq. (13):
      $$\text{In}_j^i(k+1) = \text{Out}^i_{\text{pred}(i,j)}(k+1-\delta_{\text{pred}(i,j),j}). \tag{37}$$
      According to Property 1, $\delta_{\text{pred}(i,j),j} \geq -1$. Then,
      - If $\delta_{\text{pred}(i,j),j} \geq 0$: We have that $\forall k' < k, d' \in \mathbb{N}, P(k',d')$ is true, in particular (P4) so
        $$\text{In}^i_{\text{pred}(i,j)}(k-\delta_{\text{pred}(i,j),j}) = \text{Out}^i_{\text{pred}(i,j)}(k+1-\delta_{\text{pred}(i,j),j}). \tag{38}$$
        Also (P1) is true then, $\text{In}^i_{\text{pred}(i,j)}(k-\delta_{\text{pred}(i,j),j}) \leq \tilde{\alpha}^i(T)$. Then,
        $$\text{In}_j^i(k+1) = \text{In}^i_{\text{pred}(i,j)}(k-\delta_{\text{pred}(i,j),j}) \leq \tilde{\alpha}^i(T). \tag{39}$$
      - If $\delta_{\text{pred}(i,j),j} = -1$: then $\text{Out}^i_{\text{pred}(i,j)}(k+1-\delta_{\text{pred}(i,j),j}) = \text{Out}^i_{\text{pred}(i,j)}(k+2)$. As (P4) of P(k, d-1) is true, we have
        $$\text{Out}^i_{\text{pred}(i,j)}(k+2) = \text{In}^i_{\text{pred}(i,j)}(k+1). \tag{40}$$
        Also, (P1) is true so
        $$\text{In}_j^i(k+1) = \text{In}^i_{\text{pred}(i,j)}(k+1) \leq \tilde{\alpha}^i(T). \tag{41}$$
  - P2: By defintion of B,
    $$B_{j,k\,\%\,2}(k) \overset{(11)}{=} B_{j,k\,\%\,2}(k-1) - \sum_{i \in Fl(j)} \text{Out}_j^i(k) \tag{42}$$
    $$\overset{(12)}{=} B_{j,k\,\%\,2}(k-2) + \sum_{i \in Fl(j)} In_j^i(k-1) - \sum_{i \in Fl(j)} \text{Out}_j^i(k). \tag{43}$$
    But, P(k-1) is true and in particular (P2) then, $B_{j,k\,\%\,2}(k-2) = 0$.
    Also, (P4) is true then, $\text{In}_j^i(k-1) = \text{Out}_j^i(k)$. Then, $B_{j,k\,\%\,2}(k) = 0$.
  - P3: By defintion of B,
    $$B_{j,k\,\%\,2}(k+1) \overset{(12)}{=} B_{j,k\,\%\,2}(k) + \sum_{i \in Fl(j)} In_j^i(k+1). \tag{44}$$
    According to P2, $B_{j,k\,\%\,2}(k) = 0$ and P(k) is true with in particular (P1) then,
    $$B_{j,k\,\%\,2}(k+1) \overset{(P2)}{=} \sum_{i \in Fl(j)} In_j^i(k+1) \tag{45}$$
    $$\overset{(P1)}{\leq} \sum_{i \in Fl(j)} \tilde{\alpha}^i(T) \tag{46}$$
    $$\overset{(8)}{\leq} R_j(T-2S) - \overline{Bl}_j. \tag{47}$$
  - P4: From eq. (8), (P1) and (1) it directly comes
    $$\sum_{i \in Fl(j)} In_j^i(k) \leq R_j(T-2S) - Bl_{j,k}, \tag{48}$$
    that, knowing that $B_{j,k+1\%2}(k) = 0$, Prop. 2 can be applied, and $\text{Out}_j^i(k+2) = \text{In}_j^i(k+1)$.

So $P(k,d)$ holds.

By induction, $P(k,d)$ is true forall $(k,d) \in \mathbb{N}^2$, and since $P(k,d)$ is a property stronger than the statement of the theorem then, it is proved. $(\forall (j,k,i) \in (\mathcal{P}, \mathbb{N}, \mathrm{Fl}(j)), \mathrm{Out}_j^i(k+1) = \mathrm{In}_j^i(k)$ so, the system has a Large Enough Cycle).

A.5 Proof of the necessary condition

We will prove this condition by contraposition. Let us suppose it exists $j \in \mathcal{P}$ such that

ntrapositive}

$$\sum_{i \in Fl(j)} \tilde{\alpha}^i(T) > R_j(T-2S) - \overline{Bl}_j. \tag{49}$$

Let us construct a counter example. Let $k \in \mathbb{N}$ be a cycle such that for any CQF flow i passing through j, k is greater than the length of the path between the source and this port j.

Then, we assume that each CQF flow generates a load of $\tilde{\alpha}^i(T)$ during each cycle.

As all the previous ports own a Large Enough Cycle, we have that the quantity of data from the CQF flows passing to j at cycle k is equal to the sum of each load of the CQF flows i.e.,

$$\sum_{i \in Fl(j)} In_j^i(k) = \sum_{i \in Fl(j)} \tilde{\alpha}^i(T).$$

We also assume that a lower priority flows with the maximum size arrive at just before the the CQF flows can be transmitted, at $kT + S - \varepsilon$. Then, the output of the lower priority flows at the cycle $k+1$ is $Out_j^{\mathrm{LP}}(k+1) = \overline{L}^{\mathrm{LP}}$.

Finally, we assume that on this port j, a quantity of higher priority flows of load $\overline{Bl}_j - \overline{L}^{\mathrm{LP}}$ arrives just after the lower priority packet ends its transmission i.e., at $kT + S + \frac{\overline{L}^{\mathrm{LP}}}{R_j} - \varepsilon$. Then, the output of the higher priority flows at the cycle $k+1$ is $Out_j^{\mathrm{HP}}(k+1) = \overline{Bl}_j - \overline{L}^{\mathrm{LP}}$.

However, the number of bits of the CQF flows leaving the port j in cycle $k+1$ is bound by the maximum quantity of data which can leave the port j minus the lower and higher priority flows i.e,

$$Out_j^{CQF}(k+1) = \sum_{i \in Fl(j)} Out_j^i(k+1) \tag{50}$$

$$\leq R_j(T-2S) - Out_j^{\mathrm{HP}}(k+1) - Out_j^{\mathrm{LP}}(k+1) \tag{51}$$

$$= R_j(T-2S) - \overline{Bl}_j. \tag{52}$$

According to the proposition: Equation 49,

$$\sum_{i \in Fl(j)} In_j^i(k) > \sum_{i \in Fl(j)} Out_j^i(k+1) \tag{53}$$

Then, it exists a flow i such that $In_j^i(k) > Out_j^i(k+1)$ and the cycle is not large enought.

Consequently, if the network has Large Enough Cycle then for all port j,

$$\sum_{i \in Fl(j)} \tilde{\alpha}^i(T) \leq R_j(T-2S) - \overline{Bl}_j. \tag{54}$$

APPENDIX B
PROOF OF DEFINITION/PROPERTY 1

Proof. If $T_j^{\mathrm{opt}} = 0$, it does not satisfy Theorem 1. Let $T_j^{\mathrm{opt}} > 0$. Let $(t_n)_{n \in \mathbb{N}}$ be a decreasing sequence such that $\forall n \geq 0, t_n \in \mathcal{T}_j$ and $\lim_{n \to \infty} t_n = T_j^{\mathrm{opt}}$. We have

$$\sum_{i \in Fl(j)} \tilde{\alpha}^i(T_j^{\mathrm{opt}}) \leq \sum_{i \in Fl(j)} \tilde{\alpha}^i(t_n) \leq R_j(t_n - 2S) - \overline{Bl}_j$$

where the left inequality is because $\tilde{\alpha}^i()$ is wide-sense increasing and the right inequality is because $t_n \in \mathcal{T}_j$. Thus $\sum_{i \in Fl(j)} \tilde{\alpha}^i(T_j^{\mathrm{opt}}) \leq R_j(t_n - 2S) - \overline{Bl}_j$. Taking the limit gives

$$\sum_{i \in Fl(j)} \tilde{\alpha}^i(T_j^{\mathrm{opt}}) \leq R_j(T_j^{\mathrm{opt}} - 2S) - \overline{Bl}_j$$

which shows that $T_j^{\mathrm{opt}} \in \mathcal{T}_j$. □

APPENDIX C
PROOF OF DEFINITION/PROPERTY 2

Proof. It follows immediately from the definition that if $T \in \mathcal{T}_j^{\mathrm{safe}}$ and $T' \geq T$ then $T' \in \mathcal{T}_j^{\mathrm{safe}}$. It follows that $\mathcal{T}_j^{\mathrm{safe}}$ is an interval, of the form either $\left(T_j^{\mathrm{safe}}, +\infty\right)$ or $\left[T_j^{\mathrm{safe}}, +\infty\right)$. Using the same arguments as in Appendix B, we obtain that $T_j^{\mathrm{safe}} \in \mathcal{T}_j$, hence $T_j^{\mathrm{safe}} \in \mathcal{T}_j^{\mathrm{safe}}$, and thus only the latter form is possible. □

APPENDIX D
PROOF OF DEFINITION/PROPERTY 4

Proof. The proof that $T^{\mathrm{opt}} \in \mathcal{T}$ uses the same arguments as the proof in Appendix B. Since $T^{\mathrm{opt}} \in \mathcal{T}_j$, it follows that $T^{\mathrm{opt}} \geq T_j^{\mathrm{opt}}$ for every j, hence $T^{\mathrm{opt}} \geq \max_{j \in \mathcal{P}} T_j^{\mathrm{opt}}$. The fact that the inequality is strict in general follows from the example in Figure 10. □

APPENDIX E
PROOF OF DEFINITION/PROPERTY 5

Proof. The proof that $\mathcal{T}^{\mathrm{safe}} = \left[T^{\mathrm{safe}}, +\infty\right)$ is similar to Appendix C.

Let $T^{\max} = \max_{j \in \mathcal{P}} T_j^{\mathrm{safe}}$. We have to prove $T^{\mathrm{safe}} = T^{\max}$.

On one hand, as $\forall j \in \mathcal{P}$, $T^{\max} \geq T_j^{\mathrm{safe}}$ and each T_j^{safe} is margin safe, then $T^{\max} \in \bigcap_{j \in \mathcal{P}} \mathcal{T}_j$ and it is a margin-safe cycle time for each CQF port. Consequently, $\forall t \geq T^{\max}, t \in \bigcap_{j \in \mathcal{P}} \mathcal{T}_j$, and by definition of the infimum, $T^{\max} \geq T^{\mathrm{safe}}$.

Conversely, for any fixed j, $\mathcal{T}^{\mathrm{safe}} \subset \mathcal{T}_j^{\mathrm{safe}}$ hence $T^{\mathrm{safe}} \geq T_j^{\mathrm{safe}}$. Since this holds for every j, it follows that $T^{\mathrm{safe}} \geq T^{\max}$.

□

APPENDIX F
PROOF OF PROPERTY 1

Proof. We want to prove that if $\sum_{i\in Fl(j)}\tilde{\alpha}^i$ is concave, $\mathcal{T}_j = \left\{t\in\mathbb{R}^+ \mid \sum_{i\in Fl(j)}\tilde{\alpha}^i(T) \leq R_j(T-2S) - \overline{Bl}_j\right\} = [T^{\text{opt}},+\infty)$.

We focus on the complementary and will prove that if there is a solution, then the set ${}^c\mathcal{T}_j = \left\{t\in\mathbb{R}^+ \mid \sum_{i\in Fl(j)}\tilde{\alpha}^i(T) > R_j(T-2S) - \overline{Bl}_j\right\}$ is an upper bounded interval, *i.e.* equals to $[0,T^{\text{opt}})$.

Let $T_1,T_2\notin\mathcal{T}_j$. As $\sum_{i\in Fl(j)}\tilde{\alpha}^i$ is concave, $\forall p\in[0,1]$ we have $pT_1+(1-p)T_2\in[T_1,T_2]$, and

$$\begin{aligned}\sum_{i\in Fl(j)}\tilde{\alpha}^i(pT_1+(1-p)T_2) \geq\\ p\left(\sum_{i\in Fl(j)}\tilde{\alpha}^i(T_1)\right)+(1-p)\left(\sum_{i\in Fl(j)}\tilde{\alpha}^i(T_2)\right)\\ > p(R_j(T_1-2S)-\overline{Bl}_j)+(1-p)(R_j(T_2-2S)-\overline{Bl}_j)\\ > R_j(pT_1+(1-p)T_2-2S)-\overline{Bl}_j.\end{aligned}$$

Consequently, $pT_1+(1-p)T_2\notin\mathcal{T}_j$ and ${}^c\mathcal{T}_j$ is an interval.

Now, in the specific cases where $\mathcal{T}_j=\emptyset$ or $\mathcal{T}_j=\mathbb{R}_*^+$, $T_j^{\text{opt}}=+\infty$ or $T^{\text{opt}}=0$ are respectively the lower bound of $\mathcal{T}_j$. Otherwise, there exists $\underline{T},\overline{T}\in{}^c\mathcal{T}_j$ such that ${}^c\mathcal{T}_j=|\underline{T},\overline{T}|$ where $|$ means that the interval can be open or closed or semi-closed.

Let us check if $0\in{}^c\mathcal{T}_j$: $\sum_{i\in Fl(j)}\tilde{\alpha}^i(0)=0$ and the left side term is $R_j(0-2S)-\overline{Bl}_j=-2R_jS-\overline{Bl}_j$. Then, there are different possibilites:

1) $-2R_jS-\overline{Bl}_j<0$: $\underline{T}=0$ and ${}^c\mathcal{T}_j=[0,\overline{T})$.
2) $-2R_jS-\overline{Bl}_j=0$: Using the concavity between 0 and $T\in{}^c\mathcal{T}_j$, $\forall p\in]0,1[$,

$$\begin{aligned}\sum_{i\in Fl(j)}\tilde{\alpha}^i(pT+(1-p)0) &= \sum_{i\in Fl(j)}\tilde{\alpha}^i(p\cdot T)\\ &\geq p\left(\sum_{i\in Fl(j)}\tilde{\alpha}^i(T)\right)+(1-p)\left(\sum_{i\in Fl(j)}\tilde{\alpha}^i(0)\right)\\ &\geq p\left(\sum_{i\in Fl(j)}\tilde{\alpha}^i(T)\right)\end{aligned}$$

as T is not a solution, we have

$$\begin{aligned}&> pR_jT\\ &= R_jpT.\end{aligned}$$

Consequently, $\underline{T}=0$ and ${}^c\mathcal{T}_j=[0,\overline{T})$.

Consequently, $\mathcal{T}_j\subseteq\mathbb{R}^+$ is a lower bounded interval.

Now, let us prove that this bound is reachable. Let $T_j=\inf\mathcal{T}_j$. Using the same argument as in the proof in Appendix B we have $\mathcal{T}_j=[T_j,+\infty)$. □