Thèse de doctorat

NNT: XXXXXXX

université PARIS-SACLAY

Deterministic scheduling of periodic datagrams for low latency in 5G and beyond

Thèse de doctorat de l'Université Paris-Saclay

École doctorale n° 580, Sciences et technologies de l'information et de la communication (STIC)

Unités de recherche: (1) Université Paris-Saclay, UVSQ, Données et Algorithmes pour une ville intelligente et durable, 78035, Versailles, France.
(2) Nokia Bell Labs France, 91620 Nozay, France.
Référent: : Université de Versailles-Saint-Quentin-en-Yvelines.

Thèse présentée et soutenue à ....., le .... 2021, par

Maël Guiraud

Composition du jury:

(**Johanne Cohen** Directeur de Recherche , LRI	Président/e)
Prénom Nom Titre, Affiliation	Rapporteur
Prénom Nom Titre, Affiliation	Rapporteur
(**Safia Kedad-Sidhoum** Professeur, CNAM	Examinatrice)
Dominique Barth Professeur, UVSQ	Directeur de thèse
Yann Strozecki Maître de conférence, UVSQ	Coencadrant
Olivier Marcé Ingénieur de recherche, Nokia Bell Labs France	Coencadrant
Brice Leclerc Ingénieur de recherche, Nokia Bell Labs France	Examinateur

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre: Ordonnancement periodiques de messages pour minimiser la latence dans les réseaux dans un contexte 5G et au delà

Mots clés: Cloud Radio Access Network, Ordonnancement periodique, Heuristiques de recherche locale, Analyse de complexité, Réduction de la latence, Theorie des graphes

Résumé: Cette thèse est le fruit d'une collaboration entre les laboratoires DAVID et Nokia Bell Labs France. L'idée originale est de trouver des solutions algorithmiques pour gérer des flux periodiques de manière déterministe dans les réseaux afin de contrôler et de minimiser le temps de transmission, appelé latence. L'un des objectifs de la 5G (le C-RAN, pour Cloud Radio Access Network) est de centraliser les unités de calculs des antennes radio des réseaux de télécommunications (appelé Radio Access Network) dans un même centre de calcul (le Cloud). Le réseau entre le centre de calcul et les antennes doit être capable de satisfaire les contraintes de latence imposées par les protocoles.

Nous définissions le problème de trouver un ordonnancement periodique pour les messages de façon à ce qu'ils ne se disputent jamais la même ressource, et prouvons que les différentes variantes du problème étudiés sont NP-complets. Nous étudions dans un premier temps le problème pour une topologie particulière dans laquelle tous les flux partagent un même lien. Nous proposons dans un premier temps des algorithmes polynomiaux, de plus en plus evolués, ainsi que des algorithmes FPT peremettant de trouver une solution quand le nombre de route est raisonnable, ce qui est le cas des réseaux C-RAN.

Les algorithmes développés dans cette première partie n'étant pas applicables directement aux topologies plus générales, nous proposons ensuite une forme cannonique au problème qui nous permet de définir une notion de voisinage efficace pour des heuristiques de recherches locales (descente, recherche tabou, recuit simulé). Nous utilisons cette forme cannonique pour définir un algorithme Branch and Bound efficace quand le nombre de route est modéré. Nous proposons aussi une évaluation de performance des solutions proposés par rapport aux solutions courantes de gestion des flux, et montrons que notre modèle est réalisable en pratique grace aux nouveaux equipements en cours de développement.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Title: Deterministic scheduling of periodic datagrams for low latency in 5G and beyond

Keywords: Cloud Radio Access Network, Periodic scheduling, Local search heuristics, complexity analysis, Latency reduction, Graph theory

Abstract: This thesis is the result of a collaboration between DAVID Laboratory and Nokia Bell Labs France. The original idea is to find algorithmic solutions to deterministically manage periodic flows in networks in order to control and minimize the transmission time, called latency. One of the objectives of 5G (C-RAN, for Cloud Radio Access Network) is to centralize the calculation units of the radio antennas of telecommunications networks (called Radio Access Network) in the same computer center (the Cloud). The network between the computing center and the antennas must be able to satisfy the latency constraints imposed by the protocols.

We define the problem of finding a periodic scheduling for messages so that they never compete for the same resource, and prove that the different variants of the problem studied are NP-complete. We first study the problem for a particular topology in which all the streams share the same link. We first propose polynomial algorithms of increased sophistication, and FPT algorithms that allow us to find a solution when the number of routes is reasonable, which is the case for C-RAN networks.

Since the algorithms developed in this first part are not directly adaptable to more general topologies, we then propose a canonical form to the problem which allows us to define an efficient neighborhood notion for local search heuristics (hill climbing, taboo search, simulated annealing). We use this canonical form to define an efficient Branch and Bound algorithm when the number of routes is moderate. We also propose a performance evaluation of the proposed solutions compared to current flow management solutions, and show that our model is feasible in practice thanks to new equipment under development.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France

Contents

List of Figures

Chapter 1

Mixing Periodic Datagrams and Stochastic Datagrams

In previous chapters, we presented algorithms that solve the problems of scheduling deterministic traffic in the network. In practice, networks are shared between deterministic and stochastic traffics. This is possible in practice using Time Sensitive Networking technology, that allows to manage traffics independently. The objective of this chapter is to study the impact on stochastic traffic of the algorithms we have designed to minimize the latency of deterministic traffic. We propose a method using the algorithms we have designed, to improve the latency of *all* traffics of the network.

1.1 Periodic Assignment and Random Traffic on Star Routed Networks

This section is taken from [1]. The algorithms proposed in this thesis are designed to manage deterministic periodic flows in dedicated networks. In this section, the objective is to determine the effect of adding in the network non-deterministic flows (internet traffic, best-effort) managed by statistical multiplexing.

The algorithms solving PALL are not designed to take into account best-effort traffic. In particular, they often build very compact assignments, with all messages following one another in a contention point, which is bad for the latency of best-efforts packets trying to go through the same contention point. Thus, we propose an adaptation of any algorithm solving PALL, to find assignments where the unused tics are as evenly spaced as possible to minimize the maximal latency of any random packet trying to go through the contention point.

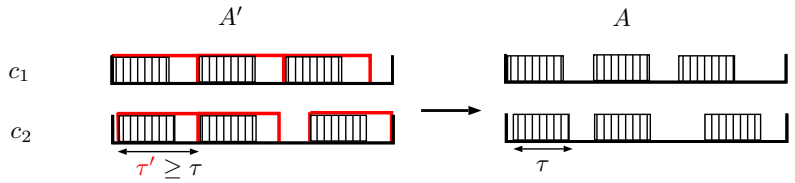

Figure 1.1 – A (P,τ')-assignment interpreted as a (P,τ)-assignment

1.1.1 Spaced Assignments

Most algorithms for PALL, when determining the waiting times, send datagrams as early as possible and thus create long sequences of datagrams in c_2, without free tics between them. We propose to modify any algorithm solving PALL on an instance with datagram size τ as follows: compute a (P,τ') assignment using the algorithm, for the largest possible $\tau' \geq \tau$.

Lemma 1. *Let $I' = (N,P,\tau',d)$ be an instance of* PALL*, for which there is an assignment, and let $\tau \leq \tau'$, then there is also an assignment for $I = (N,P,\tau,d)$.*

Proof. Let A be the assignment of I', the absence of collision is the absence of intersection between intervals $[r_i,c_1]_{P,\tau'}$ (and $[r_i,c_2]_{P,\tau'}$). If we consider A as an assignment of I, then the intervals are $[r_i,c_1]_{P,\tau}$ and are strictly included in $[r_i,c_1]_{P,\tau'}$, hence they do not have an intersection either. □

Lemma 1 gives a way to obtain a solution to the original instance from the instance with a larger message size as illustrated in Figure 1.1, with the additional property that all datagrams are separated by at least $\tau' - \tau$ free tics in each contention point. We are interested in finding the maximal τ' for which there is an assignment. Since the property of having an assignment is monotonous with regards to τ, we can do so by a dichotomous search.

We call `SPMLS`, for **S**paced `PMLS`, the adaptation of `PMLS` which finds an assignment for the largest possible τ by dichotomous search on τ. We now investigate how large are the τs for which `SPMLS` finds an assignment. In Figure 1.2, we represent the probability to find a (P,τ')-assignment function of τ'. The star routed networks are generated as in Chapter **??**, with 8 routes and length of the arcs drawn in $[P]$. The network has a load of 60% of C-RAN traffic, hence the period is set to 33,333 for $\tau = 2500$. The network is less loaded with C-RAN traffic than in the previous sections because it will also support non deterministic traffic, incurring an additional load.

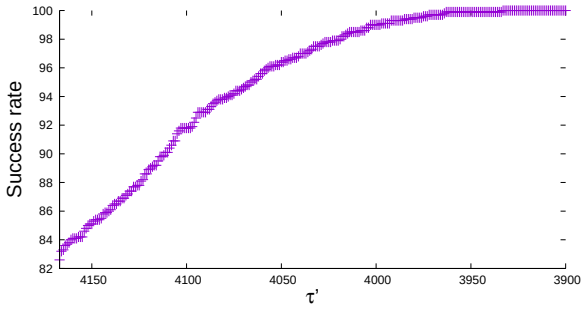

Figure 1.2 – Probability of finding a (P,τ')-assignment over 10,000 instances

For more than 80% of the instances, there is an assignment for the maximal size of a message $\tau' = \frac{P}{n} = 4166$. This means that `SPMLS` perfectly balances the free tics in the period. In the worst case, a solution with $\tau' = 3925$ is found, which still yields $3925 - 2500 = 1425$ unused tics between datagrams. Hence, we expect `SPMLS` to work well in conjunction with random traffic. The excellent performance of `PMLS` when the load is high explains this result and justifies the work we have done to solve PALL efficiently under high load rather than just requiring a mild load in applications.

1.1.2 Performance Evaluation

We evaluate in this section different ways to manage both statistical and deterministic traffics together in the same network.

1.1.2.1 Best-effort datagrams generation

Let us denote best-effort by BE. The BE traffic is generated as follows. The size of a BE datagram is small in practice, and set to 50 tics in our experiments. We generate 20% of average load of BE traffic in our experiment, to obtain a total load of 80%. The BE datagrams do not make a round trip in the network as the C-RAN datagrams, they go through a single contention point. We simulate that, by generating 20% of average load of BE datagrams for each of the two contention points c_1 and c_2. The **latency** of a BE datagram is defined as the time it must wait before going through its contention point.

On each contention point, the generation is split in two exponential distributions which

give the time before the next arrival of datagrams. The first one models background traffic, it has an average load of 15% by generating one BE datagram every 333 tics on average. The second models a burst of BE datagrams with an average load of 5%, that is, a generation of 10 datagrams, but every 10,000 tics on average.

1.1.2.2 Statistical multiplexing policy

We test several policies to deal with all traffics using statistical multiplexing. The BE traffic is managed using `FIFO`, and we propose two policies to deal with C-RAN. First, all datagrams, BE or C-RAN, are stored in the same buffer and dealt with the `FIFO` policy regardless of their type. We call this policy `FIFO`.

In order to minimize the latency of C-RAN traffic, we can store the two types of datagrams in two different buffers, managed each with `FIFO`, but we prioritize the C-RAN datagrams which are always sent first. It can be technically implemented by using TSN 802.1Qbu [2], that allows to define priority class in the traffic to schedule first the traffic with the highest priority, here the C-RAN traffic. We call this policy `FramePreemption`.

We also consider the case of C-RAN traffic scheduled by `PMLS` or `SPMLS`. Then, we need to forbid the transit of a BE datagram which collides with a C-RAN datagram. Thus, in each contention point, we reserve 50 tics (the size of a BE datagram) before the arrival of a C-RAN message. Observe that it wastes some ressources and thus slightly decreases the maximal throughput and may worsen the latency of BE datagrams.

Figure 1.3 shows the cumulative distribution of the logical latency of BE datagrams, that is the probability that a BE datagram has a latency less than some value. The distribution is computed over 1000 random instances, and for each the traffic is simulated for ten periods.

If we compare `FIFO` and `FramePreemption`, we see that the latency of BE datagrams is better (1977 tics on average) with `FIFO`. It is expected, since in `FramePreemption` the C-RAN datagrams are prioritized and thus the latency of the BE datagrams is strictly worse, 3256 tics on average. However, this is a trade-off with the margin of the C-RAN datagrams, which is strictly better for `FramePreemption`: 1919 tics on average versus 5265 tics for `FIFO`.

Using a deterministic approach for C-RAN with `PMLS`, the trade-off is even stronger: the CRAN margin is down to 0, but the BE traffic is more impacted, at a latency of 4909 tics on average. This can be explained by both reservation of tics to deal with the

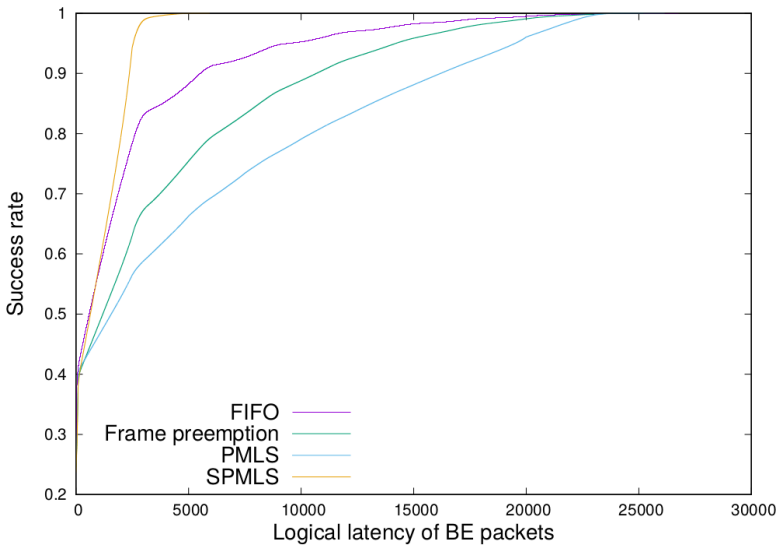

Figure 1.3 – Cumulative distribution of the latency of BE datagrams for several network management schemes

periodic sending scheme and the long sequences of CRAN datagrams without free time in contention points.

In `SPMLS`, the C-RAN traffic is smoothed over the period, in order to regularly leave some free tics for BE traffic. By construction, we still have CRAN margin of 0 but it improves the latency of BE datagrams to 949 tics on average, which is even better than with `FIFO`. This result shows that managing deterministic traffic deterministically is also good for the other sources of traffic on the network. We have already observed such a phenomenon in [3], a similar problem on an optical ring, that we describe in the next section.

1.2 Both Traffics On Optical Ring : An Industrial product

In this section, taken from [3], we study a C-RAN application based on an optical ring. We work on an industrial product which was developed in the ANR project N-GREEN described in [4, 5]. In contrast with the previous chapters, finding emission timings so that different periodic sources do not use the same resource is easy in the context of the N-GREEN optical ring with a single data-center. However, we deal with two additional difficulties arising from practice: the messages from RRHs are scattered because of the

electronic to optic interface and there are other traffics whose latency must be preserved. It turns out that the deterministic management of CRAN traffic we propose reduces the latency of CRAN traffic to the physical delay of the routes, while reducing the latency of the other traffics by smoothing the load of the ring over the period. To achieve such a good latency, our solution needs to reserve resources in advance, which slightly decreases the maximal load the N-GREEN optical ring can handle. Such an approach of reservation of the network for an application (CRAN in our context) relates to network slicing [6] or virtual-circuit-switched connections in optical networks [7, 8].

In Section 1.2.1, we model the optical ring and the traffic flow. In Section 1.2.2, we experimentally evaluate the latency when using stochastic multiplexing to manage packets insertion on the ring, with or without priority for C-RAN packets. In Section 1.2.3, we propose a deterministic way to manage C-RAN packets without buffers, which guarantees to have zero latency from buffering. We propose several refinements of this deterministic sending scheme to spread the load over time, which improves the latency of best-effort packet, or in Section 1.2.3.3, to allow the ring to support a maximal number of antennas at the cost of a very small latency for the C-RAN traffic.

1.2.1 Model of C-RAN traffic over an optical ring

N-GREEN Optical ring The unidirectional optical ring is represented by an oriented cycle. The vertices of the cycle represent the nodes of the ring, where the traffic arrives. The arcs (u,v) of the cycle have an integer weight $\omega(u,v)$ which represents the time to transmit a unit of information from u to v. By extension, if u and v are not adjacent, we denote by $\omega(u,v)$ the size of the directed path from u to v. The **ring size** is the length of the cycle, that is $\omega(u,u)$ and we denote it by RS. A **container**, of capacity C expressed in bytes, is a basic unit of data in the optical ring.

The time is discretized: a unit of time corresponds to the time needed to fill a container with data. As shown in Figure 1.4, the node u can fill a container with a data packet of size less than C bytes at time t if the container at position u at time t is *free*. If there are several packets in a node or if a node cannot fill a container, because it is not free, the remaining packets are stored in the **insertion buffer** of the node. A container goes from u to v in $\omega(u,v)$ units of time. The ring follows a **broadcast and select scheme with emission release policy**: When a container is filled by some node u, it is freed when it comes back at u after going through the whole cycle.

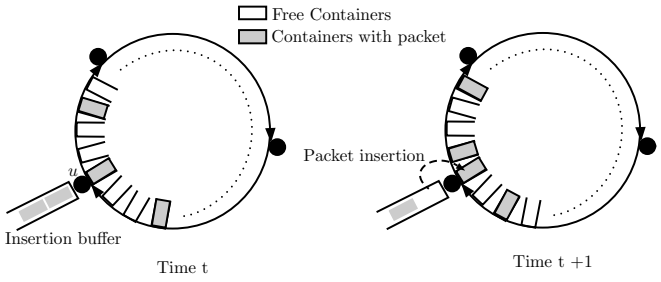

Figure 1.4 – Dynamic behavior of the ring.

C-RAN traffic The RRHs are the source of the **deterministic and periodic** C-RAN traffic. There are k RRHs attached to the ring and several RRHs can be attached to the same vertex. An RRH is linked to a node of the ring through an electronic interface of bit rate R Bps. The ring has a larger bit rate of $F \times R$ Bps. The integer F is called the **acceleration factor** between the electronic and the optical domains. A node aggregates the data received on the electronic interface during F units of time to create a packet of size C and then puts it in the insertion buffer. In each period P, an RRH emits data during a time called **emission time** or ET. Hence the RRH emits ET/F packets, i.e. requires a container of size C each F units of time during the emission time, as shown in Figure 1.5.

At each period, the data of the RRH i begins to arrive in the insertion buffer at a time o_i called **offset**. The offsets can be determined by the designer of the system and can be different for each RRH but must remain the same over all periods. We assume that all BBUs are contained in the same data-center attached to the node v. The data from u is routed to its BBU at node v through the ring and arrives at time $o_i + \omega(u,v)$ if it has been inserted in the ring upon arrival. Then, after some computation time, which w.l.o.g. is supposed to be zero, an answer is sent back from the BBU to the RRH. The same quantity of data is emitted by each BBU or RRH during any period.

The **latency** of a data packet is defined as the time it waits in an insertion buffer. Indeed, because of the ring topology, the routes between RRHs and BBUs are fixed, thus we cannot reduce the physical transmission delay of a data which depends only on the size of the arcs used. Moreover, there is only one buffering point in the N-GREEN optical ring, the insertion buffer of the node at which the data arrives. Hence, in this context, to minimize the end-to-end delay, we need to minimize the (logical) latency. More precisely, we want to reduce the latency of the C-RAN traffic to **zero**, both for the RRHs (uplink)

and the BBUs (downlink). In Section 1.2.3 we propose a deterministic mechanism with zero latency for C-RAN which also improves the latency of other data going through the optical ring. We shortly describe the nature of this additional traffic in the next paragraph.

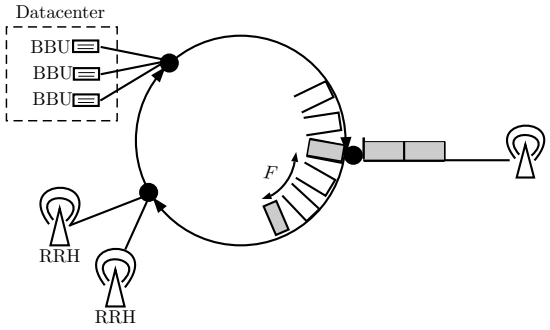

Figure 1.5 – Insertion of C-RAN traffic in the N-GREEN optical ring.

Best-Effort traffic The optical ring supports other traffics, corresponding to the internet flow. We call this traffic **Best-Effort** (BE). We want it to have the best possible distribution of latency, but since BE traffic is less critical than C-RAN traffic, we impose no hard constraint on its latency. At each node of the ring, a **contention buffer** is filled by a batch arrival process of BE data. This batch arrival process consists in generating, at each unit of time, a quantity of data drawn from a bimodal distribution to model the fact that internet traffic is bursty. Then, according to the fill rate of the contention buffer and the maximum waiting time of the data, a packet of size at most C may be created by aggregating data in the contention buffer. This packet is then put in the insertion buffer of the node. Hence, the arrival of BE messages can be modeled by a temporal law that gives the distribution of times between two arrivals of a BE packet in the insertion buffer. The computation of this distribution for the parameters of the contention buffer used in the N-GREEN optical ring is described in [9]. We use this distribution in our experiments to model arrivals of BE packets in the insertion buffer.

1.2.2 Evaluation of the latency on the N-GREEN optical ring

We first study the latency of the C-RAN and BE traffics when the ring follows an opportunistic insertion policy: When a free container goes through a node, it is filled with a packet of its insertion buffer, if there is one. Two different methods to manage the insertion buffer are experimentally compared. First, the **FIFO** rule, which consists in managing the C-RAN and BE packets in the same insertion buffer. Then, when a free container is

available, the node fills it with the oldest packet of the insertion buffer, without distinction between C-RAN and BE. This method is compared to a method called **C-RAN priority** that uses two insertion buffers: one for the BE packets, and another for the C-RAN packets. The C-RAN insertion buffer has the priority and is used to fill containers on the ring while it is non empty before considering the BE insertion buffer.

We compare experimentally these two methods in the simplest topology: The lengths of the arcs between nodes are equal and there is one RRH by node. The experimental parameters are given in Table 1.1 and chosen following [4]. In each experiment, the offsets of the RRHs are drawn uniformly at random in the period. The results are computed over 1,000 experiments in which the optical ring is simulated during 1,000,000 units of time. Fig. 1.6 gives the cumulative distribution of both C-RAN and BE traffics latencies for the FIFO and the C-RAN priority methods. The source code in C of the experiments can be found on the webpage [10].

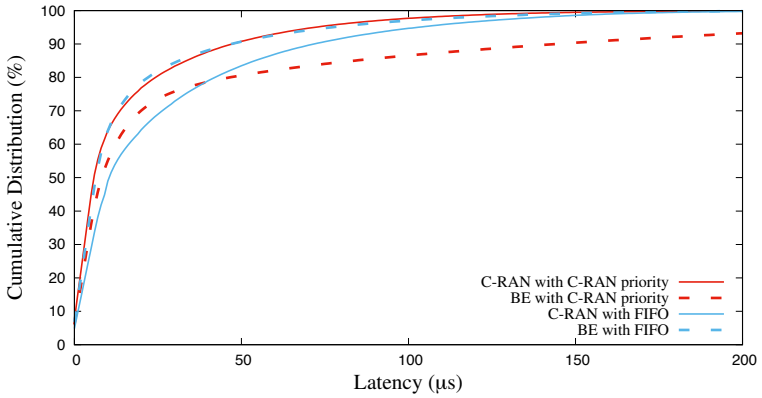

Figure 1.6 – Distribution of latencies for FIFO and C-RAN first

Bit rate of an electronic interface R	10 Gbps
Optical ring bit rate $F \times R$	100 Gbps
Acceleration factor F	10
Container size C	100 kb
Unit of time (UoT) $C/(F \times R)$	1 μs
Length traveled during one UoT	200 m

Time to go through the cycle RS	100 UoT
Emission time ET	500 UoT
Period P	1,000 UoT
Number of RRH	5
Number of nodes k	5
Load induced by C-RAN traffic	50%
Load induced by BE traffic	40%

Table 1.1 – Parameters of the N-GREEN architecture.

Unsurprisingly, the latency of the C-RAN traffic is better when we prioritize the C-RAN messages, while the BE traffic is heavily penalized. Furthermore, there is still 10% of the C-RAN traffic with a latency higher than 50μs, a problem we address in the next section.

Remark that, due to the broadcast and select mode, a message coming from any node induces the same load for all the nodes of the ring. Hence the latency of the traffics coming from any RRHs or from the BBUs are the same, which may seem couterintuitive knowing that all BBUs share the same node on the ring. This is why in Fig. 1.1 we do not ditinguish between uplink C-RAN traffic (RRH to BBU) and downlink C-RAN traffic (BBU to RRH).

1.2.3 Deterministic approach for zero latency

1.2.3.1 Reservation

Finding good offsets for the C-RAN traffic is a hard problem even for simple topologies and without BE traffic, as we have shown in previous chapters. In this section, we give a simple solution to this problem in the N-GREEN optical ring, and we adapt it to minimize the latency of the BE traffic.

Let u be the node to which is attached the RRH i. To ensure zero latency for the C-RAN traffic, the container which arrives at u at time o_i must be free so that the data from the RRH can be sent immediately on the optical ring.

To avoid latency between the arrival of the data from the RRH and its insertion on the optical ring, we allow nodes to **reserve** a container one round before using it. A container which is reserved cannot be filled by any node except the one which has reserved it (but it may not be free when it is reserved). If u reserves a container at time $o_i - RS$, then it is guaranteed that u can fill a free container at time o_i with the data of the RRH i. In the method we now describe, the C-RAN packets never wait in the node: The message sent by the RRH i arrives at its BBU at node v at time $o_i + \omega(u,v)$ and the answer is sent from the BBU at time $o_i + \omega(u,v) + 1$.

Recall that an RRH fills a container every F units of time, during a time ET. Thus if we divide the period P into **slots** of F consecutive units of time, an RRH needs to fill at most one container each slot. If an RRH emits at time o_i, then we say it is at **position** $o_i + \omega(u,v) \pmod F$. The position of an RRH corresponds to the position in a slot of the container it has emitted, when it arrives at v, the node of the BBU. If an RRH is at position p, then by construction, the corresponding BBU is at position $p+1 \pmod F$. For now, we do not allow waiting times for C-RAN traffic, hence each RRH uses a container at *the same position during all the emission time.*

Given a ring, a set of RRH's, a period and an acceleration factor F, the problem we solve here is to find an **assignment** of values of the offsets o_i's which is **valid**: two RRHs

must never use the same container in a period. Moreover we want to preserve the latency of the BE traffic. It means that the time a BE packet waits in the insertion buffer must be minimized. To do so, we must minimize the time a node waits for a free container at any point in the period, by spreading the C-RAN traffic as uniformly as possible over the period.

Figure 1.7 represents an assignment of two couples of RRH and BBU by showing the containers going through the node of the BBU during a period. Each slot has a duration of F unit of times, and, since an RRH/BBU emits a packet each F UoT during ET UoT, if we take the granularity of a slot to represent the time, the emission of a BBU/RRH is continuous in our representation, during ET/F slots. A date t in the period corresponds in Figure 1.7 to the slot t/F and is at position $t \mod F$.

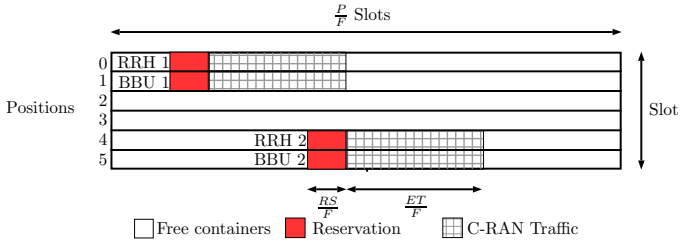

Figure 1.7 – A valid assignment with $F = 6$.

1.2.3.2 Building valid assignment with zero C-RAN latency

Remark that two RRHs which are not at the same position never use the same containers. Moreover, if we fix the offsets of the RRHs to even positions so that they do not reserve the same containers, then, because the answers of the BBU are sent without delay in our model, it will fix the offsets of the BBUs to odd positions which do not reserve the same containers. Hence, we need to deal with the RRHs only. The next proposition gives a simple method to find an assignment.

Proposition 1. *There is a valid assignment of the offsets $o_1, \ldots, o_k$ on the same position if $kET + RS \leq P$.*

Proof. W.l.o.g we fix o_1 to 0 and all the other offsets will then be chosen at position 0. Let $u_1, \ldots, u_k$ be the nodes attached to the RRHs $1, \ldots, k$. We assume that $u_1, \ldots, u_k$ are in the order of the oriented cycle. The last message emitted by the RRH 1 arrives at u_2 at time $ET - 1 + \omega(u_1, u_2)$. Therefore we can fix $o_2 = ET + \omega(u_1, u_2)$. In general we can

set $o_i = (i-1) \times ET + \omega(u_1,u_i)$ and all RRHs will use different containers at position 0 during a period. By hypothesis $k \times ET + \omega(u_1,u_1) \leq P$, thus the containers filled by the k-th RRH are freed before P. Hence, when the RRH 1 must emit something at the first unit of time of the second period, there is a free container. □

Remark that reserving free containers make them unusable for BE traffic which is akin to a loss of bandwidth. However, with our choice of emission times of the RRHs in the order of the cycle, most of the container we reserve are used by the data from some RRH. If all containers at some position are used, that is $kET + RS = P$, then there are only RS free containers wasted. In the worst case, less than $2RS$ containers are wasted by the assignment of Proposition 1.

It is now easy to derive the maximal number of antennas which can be supported by an optical ring, when using reservation and the same position for an RRH for the whole period.

Corollary 1. *There is a valid assignment with* $\lfloor \frac{P-RS}{ET} \rfloor \times \frac{F}{2}$ *antennas and zero latency.*

Proof. Following Proposition 1, the maximal number of antennas for which there is an assignment on the same position is $k = \lfloor \frac{P-RS}{ET} \rfloor$. In such an assignment, we need a second position to deal with the traffic coming from the BBUs coming back to those k antennas. Since we got F positions in the slot, the number of antennas supported by the ring is thus equal to $k \times \frac{F}{2}$. □

With the parameters of the N-GREEN ring given in Figure 1.1, we can support 5 antennas, while stochastic multiplexing can support 10 antennas albeit with extreme latency. There are two sources of inefficiency in our method. The first comes from the reservation and cannot be avoided to guarantee the latency of the C-RAN traffic. The second comes from the fact that an RRH must emit at the same position during all the emission time (to guarantee zero latency). We relax this constraint in Section 1.2.3.3 to maximize the number of antennas supported by the ring, while minimizing the loss of bandwidth due to reservation.

We now present an algorithm using reservation as in Proposition 1 to set the offsets of several RRHs at the same position. In a naive assignment, we put each RRH in an arbitrary position, for instance one RRH by position. We then propose three ideas to optimize the latency of the BE traffic, by spacing as well as possible the free containers in a period.

Balancing inside the period With the parameters of the N-GREEN ring given in Figure 1.1 ($ET = \frac{P}{2}$, $F = 10$ and $n = 5$), there are no unused position. Any assignment has exactly one BBU or RRH at each position. If all the RRHs start to emit at the first slot, then during ET there will be no free container anywhere on the ring, inducing a huge latency for the BE traffic. To mitigate this problem, in a period, the time with free containers in each position must be uniformly distributed over the period as shown in Fig. 1.8.

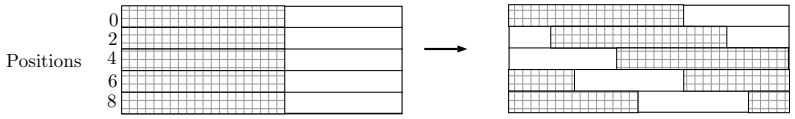

Figure 1.8 – Balancing inside the period.

Compacting positions For each position which is used by some RRH, and for each period, at least RS free containers are reserved which decreases the maximal load the system can handle. Therefore to not waste bandwidth, it is important to put as many RRHs as possible on the same position as shown in Fig. 1.9. Indeed, for any position which is not used at all, no container needs to be reserved. This strategy is also good to spread the load during the period since it maximizes the number of unused positions and for each unused position there is a container free of C-RAN traffic each F unit of times.

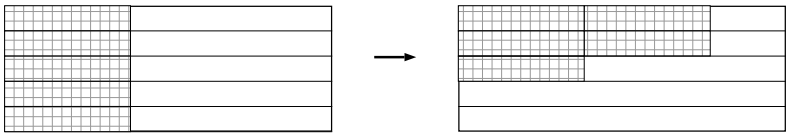

Figure 1.9 – Compacting positions.

Balancing used positions The free positions can be distributed uniformly over a slot, to minimize the time to wait before a node has access to a container from a free position, as shown in Fig. 1.10. To do so, compute the number of needed positions $x = \lceil k \times \frac{ET}{P-RS} \rceil$, with k the number of antennas using the previous strategy. Then, set the x used positions in the following way: $\lfloor \frac{F}{x} \rfloor - 1$ free positions are set between each used positions. If $\frac{F}{x}$ has a reminder r, then we set the r free remaining positions uniformly over the interval in the same way and so on until there are no more free position. It is a small optimization, since

it decreases the latency by at most $F/2$.

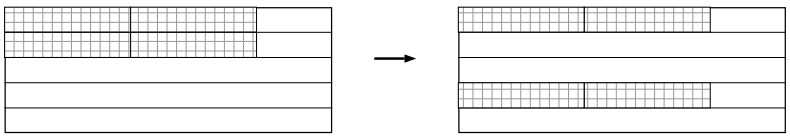

Figure 1.10 – Balancing used positions.

Experimental evaluation Our algorithm *combines the three methods* we have described to spread the load over the period. In order to understand the interest of each improvement, we present the cumulative distribution of the latency of the BE traffic using them either alone or in conjunction and we compare our algorithm to stochastic multiplexing with C-RAN priority.

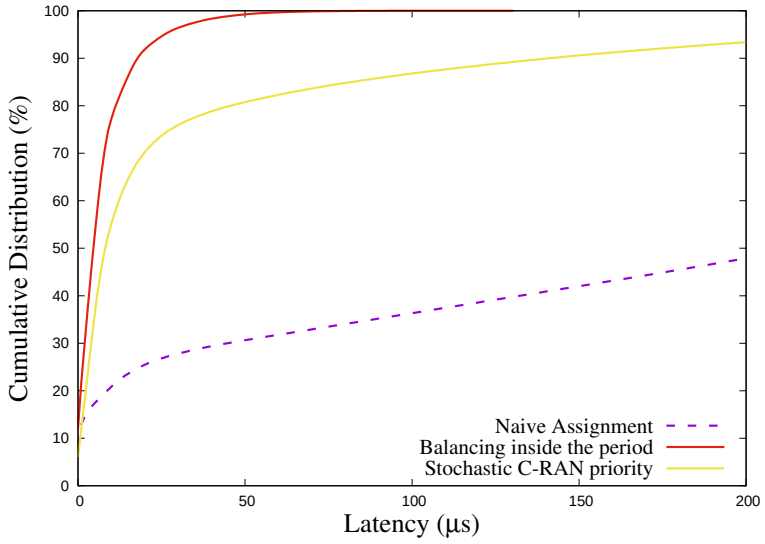

Figure 1.11 – BE latencies with a naive assignment and balancing inside the period for 5 antennas.

Figure 1.11 shows the performance of balancing the C-RAN traffic inside the period against a naive assignment in which all the RRH begin to emit at the same slot. We keep the same parameters as in Section 1.2.2 (see Table 1.1). As expected, the BE traffic latency is much better when we balance the C-RAN traffic inside the period and already much better than stochastic multiplexing.

To show the interest of compacting the positions, we must be able to put several RRHs at the same position. Hence, we change the emission time to $ET = 200$ and the number of antennas to $k = 12$ to keep the load around 90% as in the experiment of Figure 1.6. This is not out of context since the exact split of the C-RAN (the degree of centralization of the computation units in the cloud) is not fully determined yet [11].

As shown in Figure 1.12, the performance of the naive assignment is really bad. Compacting the RRHs on a minimal number of positions decreases dramatically the latency. If in addition, we balance over a period, we get another gain of latency of smaller magnitude: the average (respectively maximum) latency for BE traffic goes from 4.76μs (respectively 48μs) to 3.28μs (resp. 37μs). We did not represent the benefit of balancing used positions because the reduction in latency it yields is small as expected: the average (respectively maximum) latency for BE traffic goes from 4.76μs (resp. 48μs) to 4.43μs (resp. 44μs).

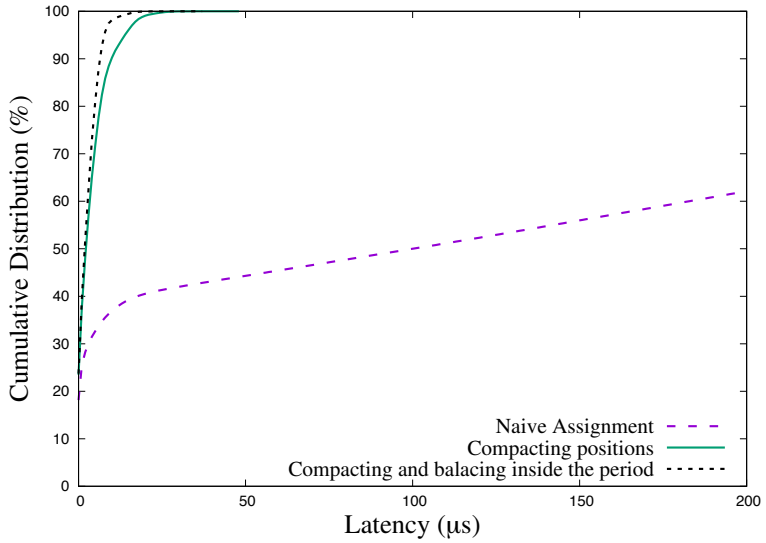

Figure 1.12 – BE latencies of compacting positions and balancing inside the period for 12 antennas.

In Figure 1.13, we compare the cumulative distribution of the latency of the BE traffic using the FIFO rule to our reservation algorithm with the three proposed improvements. The parameter are the same as in the previous experiment. The performance of our reservation algorithm is excellent, since the C-RAN traffic has *zero latency* and the BE traffic has a *better latency* than with the FIFO rule despite the cost of reservation. It is due to the balancing of the load of the C-RAN traffic over the period, that guarantee a more regular bandwidth for the BE traffic.

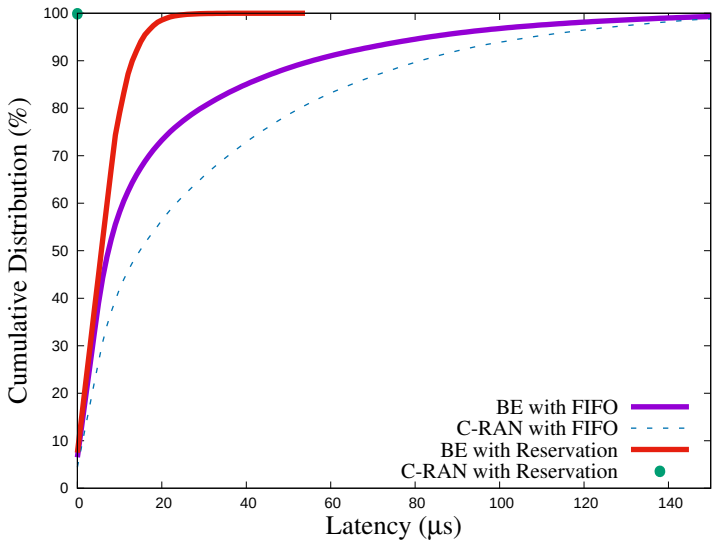

Figure 1.13 – FIFO buffer compared to the best method with reservation for 12 antennas.

1.2.3.3 Building Valid Assignments with Additional C-RAN Latency

The previous approach limits the number of antennas supported by the ring when $P - RS \mod ET \neq 0$, which is the case with N-GREEN parameters. The method we present in this section enables us to support more antennas and improves the latency of BE traffic (it reserves less free containers) by *allowing the data from an RRH to use two positions*. It is at the cost of a slightly worse latency for C-RAN traffic and it also requires in practice to implement some buffering for the C-RAN packets.

In order to support as much antennas as possible on the ring, we use *all* containers in a given position, improving on the compacting position heuristic.

Proposition 2. *There is a valid assignment for k antennas when $k \leq \lfloor \frac{P-RS}{ET} \times \frac{F}{2} \rfloor$.*

Proof. We consider the RRHs in the order of the ring. Let $l = \lfloor \frac{P-RS}{ET} \rfloor$, then we set the offsets of the first l RRHs as in Proposition 1. These RRHs are at position zero and the $(l+1)$th RRH first emits at position zero, with offset $o_{l+1} = l * ET + \omega(u_0,u_{l+1})$.

The $(l+1)$th RRH emits up to time $P - \omega(u_{l+1},u_0)$ at position zero, so that there is no conflict with RRH 0 during the next period. Hence, it has used the position zero during $x = P - \omega(u_{l+1},u_0) - l * ET - \omega(u_0,u_{l+1}) = P - l * ET - RS$. From time $P - \omega(u_{l+1},u_0) + 2$, the $(l+1)$th RRH emits at position 2 and during a time $ET - x$. Then the next RRH in

the order is assigned to position 2, and begins to emit at time $P - \omega(u_{l+1},u_0) + ET - x$ instead of zero. The rest of the assignment is built in the same way filling completely all first positions, until there are no more RRH. □

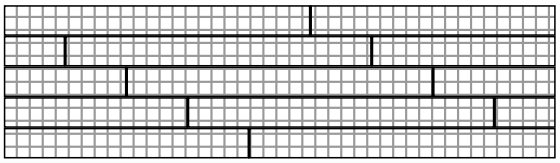

Figure 1.14 – Valid assignment for 9 antennas and the N-GREEN parameters.

Figure 1.14 illustrates the construction of Proposition 2 for the N-GREEN parameters. The loss due to reservation is exactly RS containers by used positions. Hence, it is possible to support 9 antennas (but no BE traffic in this extreme case), rather than 5 with the method of Section 1.2.3.2.

We call this new reservation algorithm **saturating positions** since it improves on compacting positions of the previous subsection. Moreover, there are no free slots in used positions, hence the idea of balancing into the period is not relevant. The only possible optimisation would be to balance the used positions, but it is not worth it since it adds additional latency for the RRHs using two different positions.

Figure 1.15 represents the cumulative distribution of the latency of BE traffic for the FIFO rule, saturating position, and balancing into the period using the N-GREEN parameters. Saturating positions reduces the BE traffic latency more than balancing into the period. This is easily explained by its lesser use of reservation. It is at the cost of a maximal latency of 2 μs for C-RAN traffic, so the designer can choose any of the two algorithms, according to the desired latency for C-RAN and BE traffic.

Conclusion

The concept of Cloud-RAN is to use non-dedicated networks, that is, networks shared with other applications. In this chapter, we study the impact of the scheduling of C-RAN flows on the latency of Best-Effort flows.

Our algorithm `ASPMLS` used to solve PALL on star routed networks tend to create long sequence of contiguous messages that monopolize the ressources during a long time. Hence, the latency of the Best-Effort flows is consequently worsen. To solve this issue, we virtually

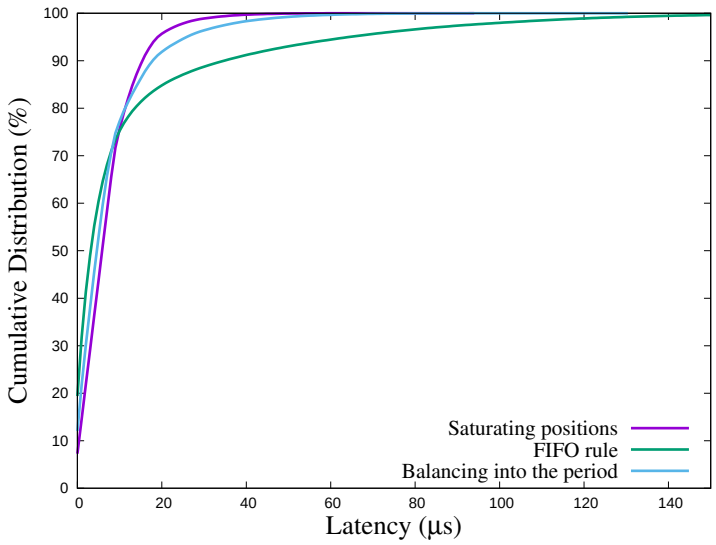

Figure 1.15 – Latencies of saturating positions, balancing into the period and FIFO rule for 5 antennas.

change the size τ of the datagram to the largest possible value τ' for which `ASPMLS` finds a solution. Then, we use the computed scheduling with the size of datagram τ, that leaves $\tau' - \tau$ free tics of time between every datagrams. Such an approach is possible because we are free to chose any offset in the period without impacting the latency. When solving SPALL, this approach would not be reasonable in term of latency, and we did not investigate yet the impact of our algorithms on Best-Effort latency.

We also present similar results on optical ring, developed for ANR project N-Green. In N-Green optial ring, the technical conception of the equipments makes the problem of scheduling the C-RAN flows trivial. We developped several techniques to smooth the load of the C-RAN flow over the period in order to let regular free tics for the Best-Effort traffic.

In both case, in order to ensure that a ressource scheduled for a route is free at the exact moment it is needed, we propose some reservation mechanisms. Reservation creates artificial use of bandwidth, which should result in lower latency for Best-Effort flows. In fact, it appears that Best-Effort latency is better when the C-RAN traffic is managed while smoothing the load on the period, even with the reservation than when all flows follows statistical multiplexing laws.

Mixing several kinds of flow and following a scheduling for a part of them is one of the major technical issue currently studied for deterministic networking. We detail in next

chapter how released standards leads us to deterministic management of the flows.

Chapter 2

Proof of Feasibility

The additional latency induced by contention buffers is one of the major manageable source of delay in a switched packets network. In this thesis, we propose algorithms that minimize contention in several topologies, either by removing the contention buffers or computing traffic organization to minimize buffering time. The objective is to delay datagrams as little as possible in the network.

Our approach of the network consists in deterministically managing datagrams in each node in order to prevent contention. When it is not possible to get rid of the contention buffers, we want to fix the duration each datagram is buffered. The contention buffers are not anymore a consequence of the traffic but a tool to manage it. This approach may look similar to the concept of Deterministic Networking. A working group from IETF called DetNet [12] works in collaboration with TSN (Time Sensitive Networking) [2], a task group of IEEE, to develop technical solutions for deterministic networking. The main difference between DetNet and TSN is the layer it focuses on. While DetNet works on Layer 3, TSN develops solutions for Layer 2. Whatever the case, using the term "deterministic" is inappropriate since all works related to DetNet or TSN are still based on statistical models. The latency guarantee given by those approaches are an upper bound on the latency, while we aim to minimize it.

In Section 2.1, we introduce the IEEE standards for TSN that allows for a better network management based on controlled management of flows in the switches. While TSN standards are designed to drive stochastic flows in network, we show that going further and managing deterministic traffic enable us to remove some technical constraint, and leads us to a new technology, that we call Hyper-TSN. Section 2.2 presents a prototype of a switch that goes beyond TSN, by delivering datagrams at exact planned dates. With Hyper-TSN, the additional latency due to contention buffers is minimized, even at full loads. Furthermore, we get rid of the synchronization constraint, with an innovative alignment mechanism. With Hyper-TSN, the latency is at physical limits. [13]

2.1 Customized Management of the Network

The model we present in Chapter ?? is based on several assumptions:

- The algorithms we develop are based on a central knowledge of the network and suppose all nodes are able to follow instructions. There must be an entity that centralizes and manages the configuration of the nodes.
- We assume the nodes are able to distinguish and manage different flows.
- Because all nodes follows a global scheduling , we assume they are able to dynamically rectify the shift between the clocks.

We explain in this chapter how the recent standards developed by TSN task group is close from such a model, and we show the limits of TSN for deterministic networking that leads us to Hyper-TSN. A detailed survey of all TSN standards can be found in [14].

2.1.1 Overview of TSN Standards

Centralized vision of the network The standard IEEE 802.1Qat SRP [15] (Stream Reservation Protocol) provides a central management framework that allows a centralized entity to collect data about the flows. It has been improved by IEEE 802.1Qcc [16]. In these standards, a centralized entity called the **controller**, collect all required information needed by users about the network (the routing, the periodicity and the size of the datagrams). This controller proposes a user interface that enable any user to collect all network information in order to compute the configuration of the network. Figure 2.1 shows a network managed by a controller, communicating via its user interface with algorithms, and able to send requirement to the nodes. Such an approach is related to Software Defined Network (SDN) [17]. We can find in [18] an example of an SDN for TSN.

Individual management of flows Standard 802.1Qbv [19] allows us to manage different flows in the nodes by a gate mechanism. Every output port of the switch is organized as follows. The flows are stored in traffic queues, and for each traffic queue, a gate is ordered to be open or closed. To do so, the switch needs a Gate Control List (GCL). This GCL is computed by the user, and sent to each switch of the network by the controller. It is a list of dates, and for every date of the list are specified the output ports (gates) of the switch which are open or closed.

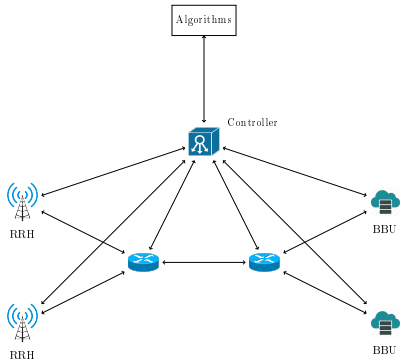

Figure 2.1 – A TSN network managed by a controller, able to collect network informations, and control the nodes behavior.

Figure 2.2 from [20] shows the mechanism of a switch using 802.1Qbv technology. Considering a given period (T_{cycle} in the figure), the switch selects at each time ($T_1, T_2, \ldots$ the queues that must be open to transmit datagrams. In figure 2.2, at time T_1, all gates except the one for scheduled traffic are open, at time T_2, all gates are closed and at time T_3 only the gate for scheduled traffic is open.

With such a mechanism, it is possible to organize the flows in order to control the latency. The GCLs are computed upstream considering the traffic (size of the datagrams, periodicity, or average throughput of each flow for non-deterministic traffic). Several works on Time Aware Shaping have been developed on this topic: [21] introduce how to manage one scheduled flow and one best-effort flow (non-periodic generation of data, stochastic model). This paper shows that by correctly setting the GCL of the nodes, it is possible to ensure no contention for one scheduled flow. Nevertheless, works about managing several scheduled flows together are mainly based on linear programming [22, 23, 24, 25], which has an high complexity and does not scale well with number of routes and contention depth of the networks.

Synchronization To be efficient, the components of the network must be completely synchronized. Such an hypothesis seems unrealistic. Standards like IEEE 802.1AS [26], or IEEE 1588 [27] propose good solutions for clock synchronization, but this problem is still difficult to solve. Indeed, even if those protocols propose good solution to re-synchronize clocks, there may be some shift between clocks of different switches of the network. The major source of shift is the need to precisely evaluate the length of a link between two components. This value can be deduced from the travel time of datagrams between two

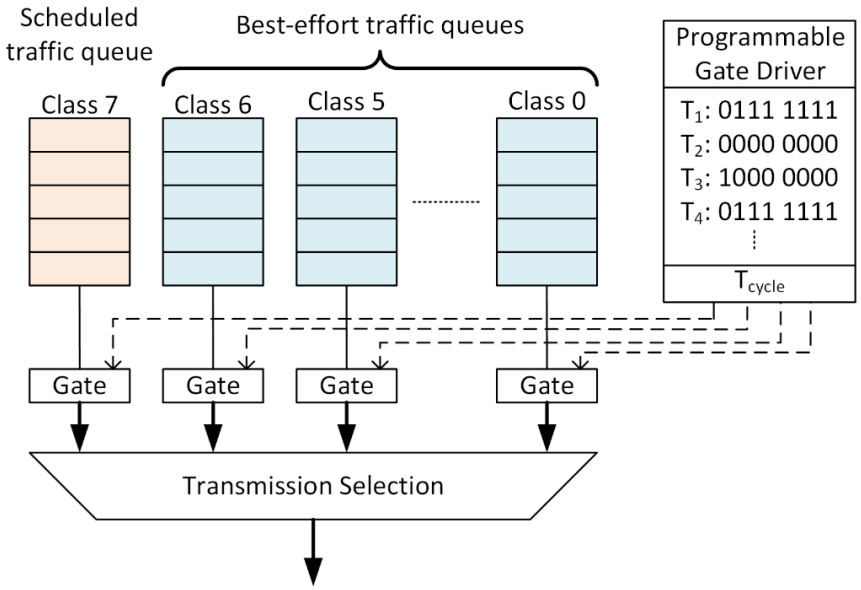

Figure 2.2 – IEEE 802.1Qbv mechanism ([20])

switches, but this value highly depends on the temperature of the link which is not constant. Network designers use *Guard Time* to prevent this problem. The GCL are computed such that a gate is open before and after the theoretical arrival of datagrams. Such a mechanism induce an artificial use of bandwidth, because a gate is open even when no datagram is being transmitted.

2.1.2 Limits of TSN when managing Deterministic flows

All the previously mentioned standards are designed for a statistical management of the flows. In this thesis, we work on deterministic flows: our models and algorithms do not rely to a statistical approach like current traffic shaping, but a deterministic approach. We compute the exact date at which each datagram is able to reach each node, without loss of bandwidth due to guard time. Furthermore, a statistical approach does not allow to control the contention and the exact forwarding time of the datagram in the network. This implies that a datagram sent periodically does not always have the same latency over time. The variation of the latency is called **jitter**. The jitter is an indicator of the stability of the network. The higher the jitter, the higher the maximal value of the latency is.

A deterministic approach requires to rethink the network management. We control the position of a datagram at every moment in order to let them pass through the nodes

without contention thanks to a gate mechanism similar to TSN Qbv. We thus need the nodes to be perfectly synchronized otherwise it could be counterproductive. Indeed, if a datagram of a flow arrives in a node before the planned date, the gate is closed and the datagram is buffered, that induces additional latency due to contention buffer. In the worst case in which a datagram arrives after the planned date, it is buffered until the gate get open. In our context of periodic flows, the datagram will be buffered between up to one period.

Also, switches in the physical layer of the network induce an additional latency due to physical buffering. In store-and-forward concept [28], datagrams are stored at reception of a node before being forwarded. However, solution like cut-through [29] allows to reduce storage size and corresponding delay to the header size only. But this is effective if the egress port is available to forward the datagram at the same time only. If not, the datagram is buffered until the port is free. Even if it is possible to adapt our model to take into consideration the physical buffering cost, it still induces additional latency, which is not desirable.

Next section present a new kind of switch, called Hyper-TSN switch, that allows to overcome all the above limits.

2.2 Deterministic management for a deterministic latency

Deterministic Networking are mentioned in the same survey cited ahead for TSN [14]. The researches about DetNet are until now limited either to linear programming for finding scheduling policies for the network -as mentioned above- or traffic shaping (see [30] for a comparison of actual traffic shaping methods). Traffic shaping methods are based on stochastic models, with bounds on the arrival of the flows, that allows to bound the maximal latency. Nevertheless, as we mentioned ahead, Deterministic Networking do not propose a deterministic management of deterministic flows in order to ensure a minimal latency and 0 jitter.

In this thesis we remove the contention buffer, or, if this is not possible, we use the contention buffers as a tool, by controlling the buffering time of each datagrams while minimizing it. Furthermore, we remove jitters in network, which is a deeper aspect of deterministic networking. Here, because of our desire to manage deterministic flows, multiple standards of TSN are not useful yet. Indeed, since the exact date of arrival of all

datagrams are computed, we can get rid of several tools designed to manage the traffic on a statistical manner.

We present in this section an Hyper-TSN switch that solves all the problematic of TSN when managing deterministic traffic. This technology is under an advanced phase of research, and a prototype has been experimented in Nokia Bell Labs [31].

2.2.1 Hyper-TSN switch

A 2x2 Hyper-TSN switch has been developed. It is composed of a switching matrix connected to a deterministic scheduler and two 10 Gbps ethernet two ways transceivers. The switching matrix includes also a monitoring circuitry. The deterministic scheduler controls the switching matrix and is configured with a timing table. This table is similar to a 802.1 Qbv Gate Control List (GCL). It defines the periodicity of the scheduling and, for each egress ports, the planned date of arrival of the frames which are part of deterministic flows. At each of these dates the deterministic scheduler sets the switching matrix to transmit data incoming on a specified ingress port. Figure 2.3 shows an example of scheduling both deterministic (but not periodic) and stochastic traffic. In such a switch, the buffers are totally absent for managed traffic: when a datagram arrives in the switch, it is instantly transmitted to the egress port, and there is no buffering operation. This process allows us to reduce the physical delay to its lowest for scheduled traffic, while it is still possible to buffer the Best-Effort traffic, which has not critical latency constraints.

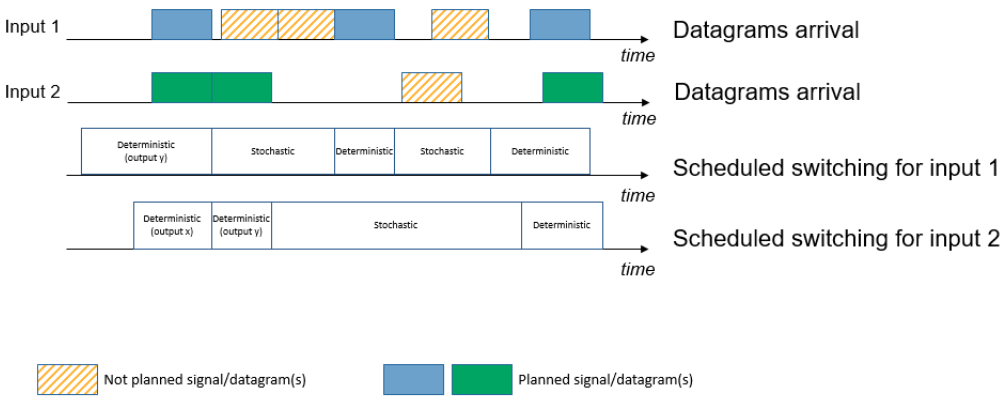

Figure 2.3 – The scheduling of a 2x2 switch on which both deterministic and stochastic traffics arrives. The deterministic traffic is forwarded without contention.

To coordinate switches, Hyper-TSN considers the flow as a reference [32] as we now explain. Since the exact arrival time of a datagram is known, if one datagram arrives before or after this expected date, this means the physical transmission delay between the sender of the datagram and the switch known by the controller is false. The monitoring circuitry of the switch detects this issue and an alert signal is sent to the controller that is able to reschedule correctly the GCL. This ensures a dynamical clock alignment between the nodes, and it is possible because the switches are developed on components with industrial clocks (Xilinx FPGA boards : Zynq UltraScale MPSoC zcu102, Zynq-7000 SoC zc706). The switch also includes a frame analyzer that enables to check that the switched frames are not corrupted and none is missing.

2.2.2 Implementation and reliability tests

To perform experiments, a generator of deterministic flows has been developed. This generator set the dates it sends the frames according to the controls received from the monitoring circuitry. The period are defined in the timing table. The size of the frames is set to fully load the ethernet links (i.e. 100% load). When starting, the monitoring circuitry detects that frames do not arrive at the planned date and sends control commands to the generator. These first frames are lost. Then, the generator corrects the dates it sends the frames, and no more shifting has been observed during the running of a 2 hours experiment. 100% of the frames are correctly switched without being corrupted or lost. The switching of each frame from the ingress port to the planed egress port is performed introducing only one clock cycle delay (here 3,87 ns). Figure 2.4 shows the Hyper-TSN 2x2 switch used for our experiments.

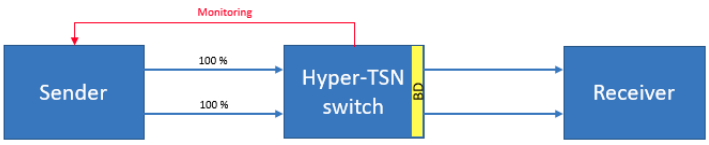

Figure 2.4 – An Hyper-TSN switch with a 2x2 switching matrix.

Conclusion

The algorithms developed in this thesis are based on several assumptions. First, we consider that a central entity is able to collect informations about the flows and the routes of the network. This is possible in practice by using SDN, which is one of the major solution for dynamic programmable networks.

The nodes need to be able to differentiate flows and to follow a scheduling to forward them. The norm TSN 802.1Qbv allows such a mechanism. We also assume that the nodes are synchronized with the precision of a tic. Even with advanced synchronization protocols, it is hard to synchronize devices with such a precision, and at least the header of the messages needs to be buffered to be read in classical packet switched networks.

As a solution to all these requirements, we present Hyper-TSN switches, which are currently in development. Those switches are based on a new vision of the network, by considering the flow as a reference. Because we manage the arrival time of every datagram in the nodes, we are able to detect a time shift if a datagram does not arrive a expected date. This allow a precise clock alignment of the devices. Furthermore, those switches are able to forward the datagrams without any buffer due to physical operations, which is an innovation in packet switched network.

Conclusion

In this thesis, we presented the problem of minimizing latency of periodic flows in a packet switched network. Current networks in use for telecommunication are based on statistical multiplexing: the links are dimensioned considering the average bit rate of the flows so that the flows can share a link most of the time or be buffered until enough capacity is available. Statistical multiplexing is a low-cost solution to deploy a network, but it does not guarantee the latency of the packets using it. If a burst of data is sent by one flow, shared resources become critical and some packets are buffered while waiting for their availability. These buffers are called contention buffers, and are a major source of latency.

We study the Cloud-RAN application case. In C-RAN, radio antennas periodically send packets to datacenters, that compute an answer and send it back to the antennas. The packets must have an end-to-end latency lower than a maximal value, required by the protocols. Statistical multiplexing is not able to guarantee an end-to-end latency for packets, and because of contention buffers, the more loaded is a network, the largest is the latency. In our C-RAN use case, the flows are periodic and a large amount of data is sent at each period by the antennas and the datacenters. Thus, managing the packets with statistical multiplexing is not appropriate.

Several working groups (DetNet, TSN, see Chapter 2), have developed standards and mechanisms ensuring an upper bound on the latency in packet switched networks. The network devices can reserve a port during a given time to forward the traffic of a given flow without contention buffer. The arrival date of the packets in a device of the network must then be known and precise. To do so, a scheduling of every output port of the devices is computed ahead. Current approaches to compute this scheduling are based on stochastic laws, since most of the internet traffic follows a stochastic behavior. In such a situation, it is impossible to completely get rid of contention buffers. Nevertheless, since our flows are deterministic (the amount of data and the periodicity of the packets remain the same all over time), we show that deterministic scheduling guarantees a minimal latency of deterministic flows and it also helps to reduce the latency of stochastic best-effort flows. Furthermore, TSN and DetNet mechanisms induce several sources of additional latency, like guard time around packets to prevent the time shift between clocks or buffering time of the header of the packets in every switch to read the destination. In this thesis, we go further by

proposing solutions to reduce the end-to-end latency of packets to its physical transmission time by proposing a new generation of switch that get rid of technical constraints imposed by a statistical vision of the network.

This thesis focuses on the problem of computing a deterministic scheduling for periodic flows, a problem that we prove to be NP-hard for arbitrary networks. We study in Chapters ??,?? this problem when the flows are unsynchronized, that is, we can choose the emission date of the packets in the sources of the flows. This case does not perfectly match with Cloud-RAN, but it corresponds to various use cases, like industry 4.0, autonomous vehicle... In Chapter ??, we give several greedy algorithms and one FPT algorithm that allows to reduce the latency to the physical transmission time (i.e. without any contention) on a common network topology with a single shared link, when the load induced by the deterministic traffic is low enough. We experimentally show using the exact FPT algorithm, that on very loaded networks (when the load is greater than 80%), it is not possible to get rid of contention buffers. We then propose solutions that buffer packets in nodes of the network, but we try to minimize this additional latency. Remark that in such an approach, the buffers are not anymore a consequence of contention that we cannot control or predict, but a tool to organize the packets. Chapter ?? study the problem of organizing flows in a network with a single shared link (as in Chapter ??), and allowing one buffer (positioned in the datacenters) on the route for every packet. We propose several greedy algorithms and one FPT algorithm based on a classical scheduling algorithm that we adapted for periodicity. The performances of our algorithm are excellent, we show it is possible to reduce latency to the physical transmission time of the longest route in 99,9% of the cases, while statistical multiplexing, even prioritizing critical flows adds a latency to the flows, due to contention buffers, equal to 1/4 of the period.

We study in Chapter ?? the C-RAN use case, in which all antennas send their messages at the same date, on arbitrary networks. We propose a compact form of the solutions to our problem, that allows to define a neighborhood of a solution. Then, several local search heuristics are designed using this notion of neighborhood. A branch and bound algorithm based on the compact form of the solutions is also proposed and run efficiently for small C-RAN network with ten to twenty routes. Then, we experimentally show that our approach dramatically over performs statistical multiplexing in terms of latency.

We then show in Chapter 1 how to adapt our algorithm to not impact best-effort flows latency while scheduling the C-RAN traffic. We explain how to adapt all our algorithms,

by solving instances with artificially increased size of messages, in a way which does not impact the latency of C-RAN datagrams, and smooth the load of C-RAN traffic all over the period. We show that, even if our approach induces an additional use of bandwidth due to resource reservation, we are able to improve the average latency of best-effort traffic while minimizing the C-RAN traffic latency. We also show similar results in an industrial optical ring, in which scheduling the C-RAN packets is trivial because of the multiplexing of the electronic signals over the optical ring.

We have proposed solutions to minimize the end-to-end latency in various use cases: Cloud-RAN, Industry 4.0, motion control, autonomous vehicle, etc... Reducing the transmission latency allows to:

- Respect latency constraints required by protocols
- Increase the Network Quality of Service
- Allow more time to the other components of the chain (computation in datacenters for C-RAN example)
- Lengthen the physical links, which means, for C-RAN; a wider area of development and thus lower exploitation and development costs (CAPEX, OPEX).

Limits and Further researchs

Several questions remain open in our work. We conjecture that PAZL and PALL are NP-hard on star routed networks, but are not yet able to prove it. The algorithms we developed for PAZL and PALL are designed for star routed networks and most of them are not easy to extend to general networks. The greedy algorithms we proposed to solve PAZL can be adapted for arbitrary networks respecting the coherent routing property, but the they can be proved to work only for small loads. These algorithms are not usable in their current state, and we need to study them carefully. Furthermore, the FPT algorithm we give to solve MINSTRA can be adapted to solve PAZL and PALL on arbitrary networks. An experimental study has to be done to see if the approach is promising and the algorithm could be optimized for the case of unsynchronised messages.

This thesis arises in the context of SDN which aims to develop dynamical and programmable networks. In C-RAN for example, the radio network aims to be able to turn off antennas when the number of connected devices is low. Our algorithms for PAZL and

PALL compute the scheduling for all flows, and must reschedule the entire solution if a flow is removed or added to the network to ensure a minimal latency. In the case of the algorithms presented for MINSTRA, the compact form of the solutions we presented allows us to efficiently add a flow to the best solution, but not to quickly re-compute the best solution when a flow is removed. A challenge is thus to design a dynamic algorithm, which can produce a new solution quickly after a local change in the network.

The measure we try to minimize is the maximal end-to-end latency of all flows. This often means that the flow using the longest route is never delayed in contention buffer, but the other flows are more or less impacted by the solutions we give. One can imagine use-cases in which the constraint on latency is not as strict, and where it makes more sense to minimize the average latency of the flows. For this variant of the problem, the study of PAZL is still relevant, but it may be simpler from a complexiy point of view and the algorithms may be quite different from the ones presented in this thesis. We can also expand our model to allow flows in the networks, with different periods or sizes of message. Already several methods fail for mixing different message sizes, since solving the problem WTA becomes NP-hard in this context and mixing several periods require to completely change our algorithms.

In Section 1.2 we introduce the fact that links of the networks may have different capacity. This induces additional latency due to the physical conversion. It could allow use cases requiring low latency to be developed on metropolitan networks. It is possible to adapt our model to take into account this issue, and to consider this physical conversion time while computing the solutions. However, dealing with this

Last, we consider that the routing is given in our model. It could be interesting to compute the routing along with the assignment to further improve the quality of the results. We plan to study this question on star routed networks, where the number of routings is limited and an exhaustive approach is conceivable.

Bibliography

[1] Dominique Barth, Maël Guiraud, and Yann Strozecki. "Deterministic Scheduling of Periodic Messages for Cloud RAN". In: *CoRR* abs/1801.07029 (2018). arXiv: 1801.07029. URL: http://arxiv.org/abs/1801.07029.

[2] *Time-Sensitive Networking Task Group*. http://www.ieee802.org/1/pages/tsn.html. Accessed: 2021-02-10.

[3] Dominique Barth, Maël Guiraud, and Yann Strozecki. "Deterministic Contention Management for Low Latency Cloud RAN over an Optical Ring". In: *Optical Network Design and Modeling - 23rd IFIP WG 6.10 International Conference, ONDM 2019*. Vol. 11616. Lecture Notes in Computer Science. Springer, 2019, pp. 479–491.

[4] Dominique Chiaroni. "Network Energy: Problematic and solutions towards sustainable ICT". In: *invited paper, International Commission of Optics (ICO-24)* (2017).

[5] Bogdan Uscumlic et al. "Scalable deterministic scheduling for WDM slot switching Xhaul with zero-jitter". In: *2018 International Conference on Optical Network Design and Modeling (ONDM)*. IEEE. 2018, pp. 100–105.

[6] Menglan Jiang, Massimo Condoluci, and Toktam Mahmoodi. "Network slicing management & prioritization in 5G mobile systems". In: *European wireless*. 2016, pp. 1–6.

[7] Christian Cadéré et al. "Virtual circuit allocation with QoS guarantees in the ECOFRAME optical ring". In: *Optical Network Design and Modeling (ONDM), 2010 14th Conference on*. IEEE. 2010, pp. 1–6.

[8] Ted H Szymanski. "An ultra-low-latency guaranteed-rate Internet for cloud services". In: *IEEE/ACM Transactions on Networking* 24.1 (2016), pp. 123–136.

[9] Youssef Ait el mahjoub, Hind Castel-Taleb, and Jean-Michel Fourneau. "Performance and energy efficiency analysis in NGREEN optical network". In: *2018 14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob) (WiMob 2018)*. Limassol, Cyprus, Oct. 2018.

[10] *Maël Guiraud's website*. https://mael-guiraud.github.io/.

[11] China Mobile. "C-RAN: the road towards green RAN". In: *White Paper, ver* 2 (2011).

[12] Norman Finn and Pascal Thubert. *Deterministic Networking Architecture*. Internet-Draft draft-finn-detnet-architecture-08. Work in Progress. Internet Engineering Task Force, 2016. 32 pp. URL: https://tools.ietf.org/html/draft-finn-detnet-architecture-08.

[13] *ALTO Performance Cost Metrics draft-ietf-alto-performance-metrics-12*. https://tools.ietf.org/html/draft-ietf-alto-performance-metrics-12. Accessed: 2021-02-11.

[14] A. Nasrallah et al. "Ultra-Low Latency (ULL) Networks: The IEEE TSN and IETF DetNet Standards and Related 5G ULL Research". In: *IEEE Communications Surveys Tutorials* 21.1 (2019), pp. 88–145. DOI: 10.1109/COMST.2018.2869350.

[15] Daniel Bujosa, Inés Álvarez, and Julian Proenza. "CSRP: an Enhanced Protocol for Consistent Reservation of Resources in AVB/TSN". In: *IEEE Transactions on Industrial Informatics* PP (Aug. 2020), pp. 1–1. DOI: 10.1109/TII.2020.3015926.

[16] "IEEE Draft Standard for Local and Metropolitan Area Networks: Overview and Architecture". In: *IEEE P802-REV/D1.7* (2014), pp. 1–68.

[17] Yong Li and Min Chen. "Software-defined network function virtualization: A survey". In: *IEEE Access* 3 (2015), pp. 2542–2553.

[18] N. G. Nayak, F. Durr, and K. Rothermel. "Software-defined environment for reconfigurable manufacturing systems". In: *2015 5th International Conference on the Internet of Things (IOT)*. 2015, pp. 122–129. DOI: 10.1109/IOT.2015.7356556.

[19] "IEEE Standard for Local and metropolitan area networks – Bridges and Bridged Networks - Amendment 25: Enhancements for Scheduled Traffic". In: *IEEE Std 802.1Qbv-2015 (Amendment to IEEE Std 802.1Q-2014 as amended by IEEE Std 802.1Qca-2015, IEEE Std 802.1Qcd-2015, and IEEE Std 802.1Q-2014/Cor 1-2015)* (2016), pp. 1–57. DOI: 10.1109/IEEESTD.2016.8613095.

[20] Frank Dürr and Naresh Ganesh Nayak. "No-wait packet scheduling for IEEE time-sensitive networks (TSN)". In: *Proceedings of the 24th International Conference on Real-Time Networks and Systems*. 2016, pp. 203–212.

[21] Mohamad Kenan Al-Hares et al. "Modeling time aware shaping in an Ethernet fronthaul". In: *GLOBECOM 2017-2017 IEEE Global Communications Conference*. IEEE. 2017, pp. 1–6.

[22] Wilfried Steiner, Silviu S Craciunas, and Ramon Serna Oliver. "Traffic planning for time-sensitive communication". In: *IEEE Communications Standards Magazine* 2.2 (2018), pp. 42–47.

[23] Silviu Craciunas, Ramon Serna Oliver, and Wilfried Steiner. "Formal Scheduling Constraints for Time-Sensitive Networks". In: (Dec. 2017).

[24] Naresh Ganesh Nayak, Frank Dürr, and Kurt Rothermel. "Incremental flow scheduling and routing in time-sensitive software-defined networks". In: *IEEE Transactions on Industrial Informatics* 14.5 (2017), pp. 2066–2075.

[25] Naresh Nayak, Frank Dürr, and Kurt Rothermel. "Time-sensitive Software-defined Network (TSSDN) for Real-time Applications". In: Oct. 2016, pp. 193–202. DOI: 10.1145/2997465.2997487.

[26] "IEEE Standard for Local and Metropolitan Area Networks - Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks". In: *IEEE Std 802.1AS-2011* (2011), pp. 1–292. DOI: 10.1109/IEEESTD.2011.5741898.

[27] "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems". In: *IEEE Std 1588-2008 (Revision of IEEE Std 1588-2002)* (2008), pp. 1–300. DOI: 10.1109/IEEESTD.2008.4579760.

[28] Elbert G Tindell and Kyle Crawford. *Store and forward video system*. US Patent 5,130,792. 1992.

[29] Parviz Kermani and Leonard Kleinrock. "Virtual cut-through: A new computer communication switching technique". In: *Computer Networks (1976)* 3.4 (1979), pp. 267–286.

[30] S. Thangamuthu et al. "Analysis of Ethernet-switch traffic shapers for in-vehicle networking applications". In: *2015 Design, Automation Test in Europe Conference Exhibition (DATE)*. 2015, pp. 55–60. DOI: 10.7873/DATE.2015.0045.

[31] Maël Guiraud, Olivier Marcé, and Brice Leclerc. "An Experimental Platform for Hyper TSN Flows Scheduling and Control Automation". In: (). To be published.

[32] Brice Leclerc et al. *Optical packet switching based on traffic properties*. US Patent 10,701,466. 2020.