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

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 :

Prénom Nom	Président/e
Titre, Affiliation	
Prénom Nom	Rapportrice
Titre, Affiliation	
Prénom Nom	Rapporteur
Titre, Affiliation	
Prénom Nom	Examinatrice
Titre, Affiliation	
Prénom Nom	Examinateur
Titre, Affiliation	
Dominique Barth	Directeur
Professeur, UVSQ	
Yann Strozecki	Coencadrant
Maître de conférence, UVSQ	
Olivier Marcé	Coencadrant
Ingénieur de recherche	Nokia Bell Labs France
Brice Leclerc	Coencadrant
Ingénieur de recherche	Nokia Bell Labs France

Thèse de doctorat

NNT : 2020UPASA000

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.

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

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.

Table des matières

Table des figures

Présentation de la thèse

Les travaux présentés dans cette thèse s'inscrivent dans le contexte du développement de la 5G, et sont plus particulièrement axés sur la réduction de la latence dans les réseaux cœur opérateur. L'un des objectifs pour la 5G est de garantir une latence bout en bout la plus faible possible, afin de pouvoir développer des applications pour lesquelles le temps de réponse est critique (véhicules autonomes, industrie 4.0, etc.) Le cas d'application que nous étudions est le C-RAN (pour Cloud Radio Access Network). Le but du C-RAN est de centraliser les unités de calculs situés aux pieds des antennes dans un ou plusieurs centres de calculs, afin de faciliter la maintenance et de réduire les couts d'exploitations. Les antennes envoient périodiquement des messages aux centres de calculs, qui calculent une réponse et l'envoient aux antennes avec la même fréquence. Le temps écoulé entre l'envoi d'un message par une antenne et la réception de sa réponse doit être inférieur à 3 ms, une contrainte imposée par le protocole de communication radio (HARQ). Ces messages envoyés sont très lourds, et utilisent donc beaucoup de bande passante dans les réseaux.

La gestion actuelle des réseaux, le multiplexage statistique consiste à dimensionner les liens de façon à ce que le flux moyen de message utilisant ces liens puisse passer en même temps. Quand trop de paquets doivent utiliser un lien en même temps, on parle de contention et une partie d'entre eux est mise dans une file d'attente, que nous appelons buffer de contention. Faire attendre les messages dans ces buffers de contention augmente la latence des paquets. Plus un réseau est chargé, plus la chance que les latences augmentent à cause de la contention. Les réseaux C-RAN envoyant une grande quantité de messages nécessitant une faible latence sont donc incompatibles avec la notion de multiplexage statistique.

Plusieurs groupes de travail proposent aujourd'hui des solutions techniques (présentées dans le chapitre 1) pour aider à contrôler la latence dans les réseaux. Avec ces solutions, les équipements du réseau sont capables de réserver une partie des ressources à un temps donné pour un paquet donné. Il faut toutefois calculer les dates auxquels les paquets doivent arriver dans les nœuds du réseau. Cette thèse se concentre sur le fait d'organiser les paquets, de façon à ce qu'ils n'entrent pas en collisions dans aucun des nœuds, dans le but de supprimer les buffers de contention. Se passer complètement des buffers de contention n'est pas toujours possible, notamment lorsque les réseaux sont composés de beaucoup de nœuds. Dans ce cas, l'objectif de nos travaux est de minimiser le temps passé par les paquets dans les files d'attentes. Il est important de souligner que dans ce cas-là les temps de contentions ne sont plus subis comme pour le multiplexage statistique, mais contrôlés.

Nous modélisons un réseau par un multigraphe orienté pondéré dont les sommets, représentent les points de contention entre messages dans le réseau. Deux messages rentrent en conflit s'ils doivent passer par le même point de contention au même moment. Nous étudions dans un premier temps le problème sur des réseaux simples, mais communs avec seulement deux points de contention en série. Nous définissons le problème de décision consistant à choisir la date de passage de chaque message dans chacun de ces deux points de contention de façon à ce aucun message n'ai de conflit avec un autre dans le réseau. Ce problème ressemble à des problèmes classique d'ordonnancement mais l'envoi periodique de nos flux en fait un problème original et difficile. Nous prouvons dans le chapitre 2 que le problème est NP-complet, même sur des graphes de faible largeur (et profondeur) d'arbre en réduisant des problèmes de coloration d'arcs et d'arêtes au nôtre.

Nous étudions dans le chapitre 3 le problème d'organiser des flux non-synchronisés dans le réseau sans aucun buffer, c'est-à-dire de trouver une temps de départ des messages au début de leur route de façon à ce qu'ils ne soient pas en conflit avec les autres messages, sans que ce temps de départ ne soit considéré comme du temps de contention. Les solutions de ce problème sont toutes optimales en termes de latence, car aucun temps de contention n'est ajouté aux messages. Nous décrivons des algorithmes de plus en plus évolués visant à optimiser une mesure commune sur l'impact d'ajouter un message à la solution partielle calculée. Ce algorithmes nous permettent de garantir qu'une solution au problème existe quand la charge du réseau est inférieure à 40% (et même jusqu'a 61% pour des messages de taille 1). Nous proposons aussi un algorithme FPT (quand le problème est paramétré par le nombre de routes) qui nous permet de calculer la solution optimale quand le nombre de routes est inférieur à 20. Les résultats montrent que le problème ne peut pas être résolu quand la charge du réseau est supérieure à 80%. C'est pourquoi nous traitons dans le chapitre 4 le problème d'organiser les flux avec un buffer sur la route, de façon à offrir un plus grand degré de liberté. Nous étudions plus particulièrement le problème de minimisation, c'est-à-dire, trouver une solution pour laquelle la plus grande latence est minimale. Nous proposons une approche en deux parties. Les temps d'envoi des messages sur le premier point de contention sont fixés en amont et nous résolvons le problème de choisir le temps d'attente de chaque message dans le second point de contention. Pour cela, nous décrivons un algorithme polynomial basé sur le problème d'ordonnancement classique de la littérature, adapté pour la périodicité. Nous proposons aussi un algorithme FPT basé sur le même principe, mais qui garanti de trouver la solution optimale. Nous montrons que nous sommes capables de trouver des solutions pour lesquelles la latence est minimale pour 99.9% des instances dans des réseaux très chargés, et que nos travaux montrent des résultats excellents comparé au multiplexage statistique.

Dans le chapitre 5, nous étudions le problème d'organiser des flux synchronisés sur tout type de DAG. Dans ce cas, tous les messages sont envoyés en même temps par les sources et nous nous permettons d'ajouter du temps de contention à chaque point de contention du réseau. Nous étudions le problème de minimiser la plus grande latence dans le réseau. Nous commençons par décrire des algorithmes gloutons qui trouvent une solution réalisable pour n'importe quelle charge, qui servent de point de départ aux algorithmes de recherche locales utilisés ensuites. Nous définissions une forme canonique du problème qui nous permet de proposer une notion de voisinage entre les solutions afin d'explorer l'ensemble de ces dernières. Nous étudions les performances des algorithmes de recherche d'optimum local (Descente, recherche tabou, recuit simulé) et nous proposons un algorithme Branch and Bound qui énumère l'ensemble des solutions sous forme canonique afin de trouver la solution optimale. Nous montrons expérimentalement que l'algorithme Branch and Bound est capable de trouver une solution optimale en un temps raisonnable pour 12 routes, tandis que le recuit simulé permet de trouver des solutions bien meilleures que le multiplexage statistique pour n'importe quelle instance.

Nous étudions ensuite dans le chapitre 6 l'impact de nos algorithme d'ordonnancement de flux C-RAN lorsqu'ils partagent le réseau avec des flux Best-Effort. Nous proposons une méthode d'adaptation de nos algorithmes qui permet de lisser la charge des flux C-RAN tout au long de la période, sans augmenter leurs latences. Les résultats montrent qu'en organisant les flux de façon déterministe comme nous le faisons requiert d'utiliser un peu plus de bande passante pour réserver les ressources, la latence moyenne des flux Best-Effort est meilleure qu'avec le multiplexage statistique.

Toutes nos approches se basent sur des hypothèses techniques fortes : les flux doivent être parfaitement synchronisés, le réseau doit être intelligent et programmable. Le chapitre 7 fait le point sur les standards récemment développés qui se rapprochent de nos hypothèses. Nous montrons aussi les limites de ces standards, et nous introduisons un équipement en phase de développement qui nous permettrais de réduire la latence dans les réseaux au temps physique de transmission.

Introduction (in english)

The work presented in this thesis fits into the development of 5G, and is more particularly focused on the reduction of latency in core operator networks. One of the objectives for 5G is to guarantee low end-to-end latency, in order to be able to develop applications for which response time is critical (autonomous vehicles, industry 4.0, etc.). The application case we are studying is C-RAN (for Cloud Radio Access Network). The goal of C-RAN is to centralize the computing units located at the foot of the antennas in one or more data-centers, in order to facilitate maintenance and reduce operating costs. The antennas periodically send messages to the data-centers, which calculate an answer and send it to the antennas with the same frequency. The time elapsed between the sending of a message by an antenna and the reception of its answer must be less than 3 ms, a constraint imposed by the radio communication protocol (HARQ). These sent messages are very heavy, and therefore use a lot of bandwidth in networks.

Current networks management, called statistical multiplexing consists in dimensioning the links so that the average message flow using these links can use it at the same time. When too many packets have to use a link at the same time, we talk about contention and some of them are queued, in what we call contention buffer. Time spent by messages in these contention buffers increases packet latency. The more a network is loaded, the more the chance that latencies increase because of contention. In C-RAN networks, the antennas and data-centers send a large amount of messages requiring low latency. This use case is then incompatible with the notion of statistical multiplexing.

Several working groups are now proposing technical solutions (presented in chapter 1) in order to help to control latency in networks. With these solutions, network equipments are able to reserve part of the resources at a given time for a given packet. However, it is necessary to calculate the dates at which packets must arrive at the network nodes. This thesis focuses on organizing packets so that they do not collide in any of the nodes, in order to remove contention buffers. totally get rid of contention buffers is not always possible, especially when networks are composed of many nodes. In this case, the goal of our work is to minimize the time spent by packets in the queues. It is important to underline that in this case contention times are no longer undergone, as in statistical multiplexing, but controlled.

We model a network using a weighted directed multigraph whose vertices represent the contention points between messages in the network. Two messages conflict if they must pass through the same contention point at the same time. We first study the problem on simple but common networks with only two serial contention points. We define the decision problem of choosing the date at which each message through each of these two contention points so that no message has a conflict with another in the network. This problem looks like classical scheduling problems but the periodicity of our flows makes our problem original and difficult. We prove in chapter 2 that the problem is NP-complete, even on graphs with small treewidth (and depth) by reducing arc and edge coloring problems to ours.

We study in chapter 3 the problem of organizing non-synchronized flows in the network without any buffer, i.e. finding a departure time for messages at the beginning of their route so that they do not conflict with other messages, without additional contention time due to this departure time. The solutions to this problem are all optimal in terms of latency, because no contention time is added to the messages. We describe increasingly advanced algorithms aimed at optimizing a common measure of the impact of adding a message to the partial solution. These algorithms allow us to ensure that a solution to the problem exists when the network load is less than 40% (and even up to 61% for messages of size 1). We also propose an FPT algorithm (when the problem is parameterized by the number of routes) that allows us to calculate the optimal solution when the number of routes is less than 20. The results show that the problem cannot be solved when the network load is higher than 80%. This is why we deal in chapter 4 with the problem of organizing the flows with one buffer on the route, so as to offer a greater degree of freedom. In particular, we study the problem of minimization, that is, finding a solution for which the highest latency is minimal. We propose a two-stage approach. The departure times of the messages are fixed upstream such that there is no conflict in the first contention point, and we solve the problem of choosing the waiting time for each message on the second contention point. To do so, we describe a polynomial algorithm based on the classical scheduling problem of the literature, adapted for periodicity. We also propose an FPT algorithm based on the same principle,that guarantees to find the optimal solution. We show that we are able to find solutions for which latency is minimal for 99.9% of instances on highly loaded networks, and that our work shows excellent results compared to statistical multiplexing.

In chapter 5, we study the problem of organizing synchronized flows on any type of DAG. In this case, all messages are sent at the same time by the sources and we allow ourselves to add contention time at each contention point of the network. We study the problem of minimizing the highest latency in the network. We start by describing greedy algorithms that find a valid solution for any load, which serve as a starting point for the local search algorithms used as follows. We define a canonical form of the problem that allows us to propose an efficient notion of neighborhood between the solutions in order to explore all of the solutions. We study the performances of the local search algorithms (Descent, taboo search, simulated annealing) and we propose a Branch and Bound algorithm that lists all the solutions in canonical form in order to find the optimal solution. We show experimentally that the Branch and Bound algorithm is able to find an optimal solution in a reasonable time for 12 routes, while simulated annealing allows to find much better solutions than statistical multiplexing for any instance.

We then study in chapter 6 the impact of our scheduling of C-RAN flows when sharing the network with Best-Effort flows. We propose a method to adapt our algorithms that smooth the load of C-RAN flows throughout the period, without increasing their latencies. The results show that by organizing the flows in a deterministic way as we do, even if it requires a little more bandwidth to resources reservation, the average latency of the Best-Effort flows is better than with statistical multiplexing.

All our approaches are based on strong technical assumptions : the flows must be perfectly synchronized, the network must be intelligent and programmable. The chapter 7 reviews the recently developed standards that are close to our assumptions. We also show the limits of these standards, and we introduce equipment under development that would allow us to reduce latency in networks to the physical transmission time.

Chapitre 1

Algorithmic and industrial context

1.1 Industrial context

1.1.1 What is 5G ?

Telecoms networks must manage more and more users while continually improving their bandwidth, latency and reliability. Nowadays, 4G is the standard deployed in most of the territory, and 5G is the new technology under deployment. The term 5G defines a set of functional specifications. The organism that proposes these specifications is the ITU (International Telecommunication Union), an agency of the United Nation responsible for information and communications. For several years, the ITU-R (radiocommunication component of the ITU) has been working to determine the functional aspects that 5G must satisfy. Figure 1.1 from [1] illustrates some of those functional aspects, that ITU-R has formally referenced under the name IMT-2020 [2] (the requirements of 4G are referenced as IMT-advanced) : a bitrate up to 20Gbps ($\times$20 compared to 4G), low end to end latencies down to 1ms (10 times lower than in 4G, we are focusing on this aspect in this thesis). Also, 5G aims to offer an higher connection density (up to 1 million device/km^2), with an higher traffic capacity (from 0,1$Mbit/s/m^2$ in 4G to 10$Mbit/s/m^2$ in 5G) thanks to a wider use of the spectrum. Other aspects as a better energy efficiency (100 times better in 5G than in 4G) or a better mobility are required.

All these characteristics allow various application cases. Figure 1.2 taken from [3] establishes a non-exhaustive list of them, according to their different technical constraints. One of them is **low latency**, required for applications like motion control that work in real time. Also, 5G aims to develop dynamic programmable networks, for greater flexibility of use. This thesis focuses on developing algorithm to dynamically manage flows in the network in order to provide low latency.

On the other hand, a higher bandwidth is useful for applications like video streaming,

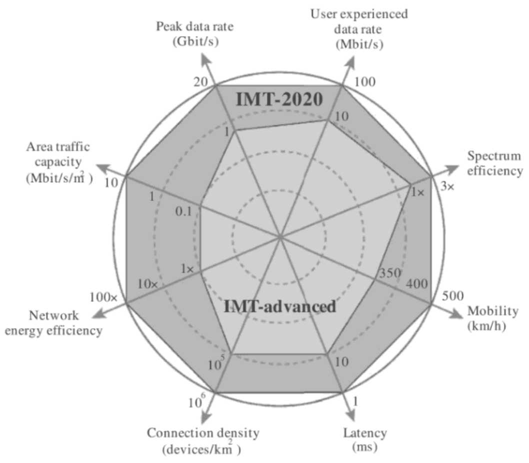

FIGURE 1.1 – 5G performances required by ITU-R ([1])

augmented reality or ensuring the connectivity of a large number of terminals. By relaxing latency and bandwidth constraints, it is possible to expand further the number of devices (up to 1 million) for applications like sensors networks.

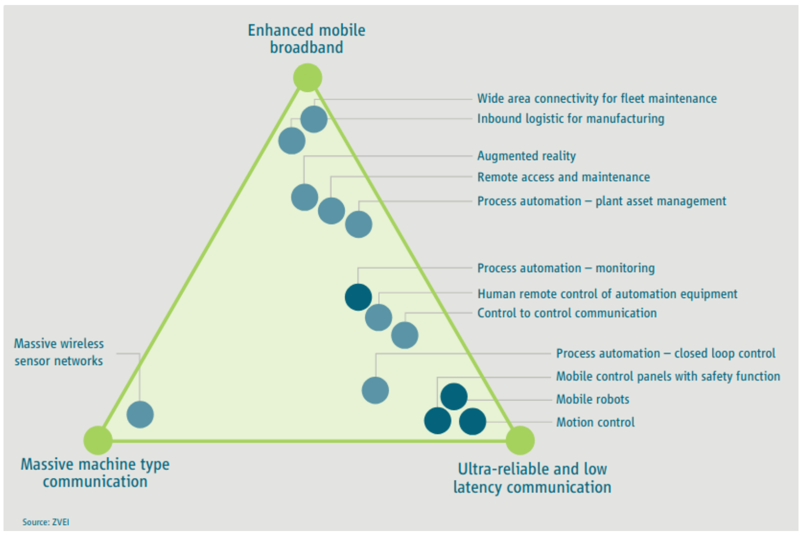
Enhanced mobile broadband
Wide area connectivity for fleet maintenance
Inbound logistic for manufacturing
Augmented reality
Remote access and maintenance
Process automation – plant asset management
Process automation – monitoring
Human remote control of automation equipment
Control to control communication
Massive wireless sensor networks
Process automation – closed loop control
Mobile control panels with safety function
Mobile robots
Motion control
Massive machine type communication
Ultra-reliable and low latency communication
Source: ZVEI

FIGURE 1.2 – Some examples of use cases for 5G ([3])

1.1.2 Which aspect of 5G do we focus on ?

To meet these 5G functional specifications, the network equipments must follow technical standards. The 3GPP (3rd Generation Partnership Project) is an union between several standard organizations which defines the technical specifications for 5G. 3GPP regularly publishes releases, that regroup new specifications. The first release focused on 5G was

Release 15 (R15), released in 2018 [4].

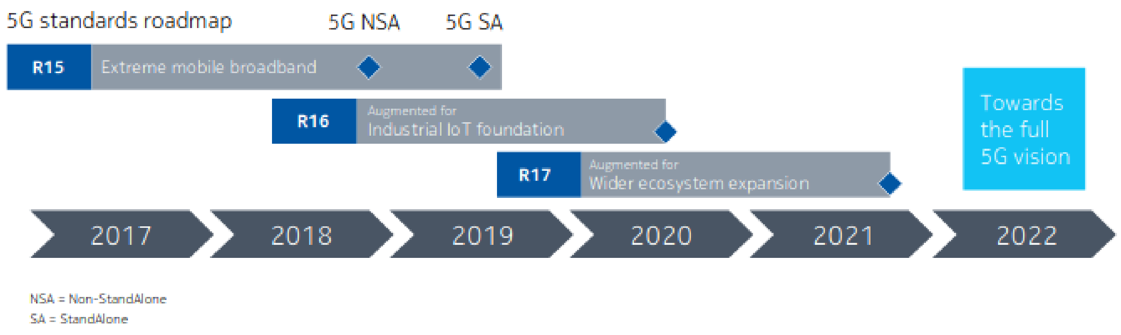

FIGURE 1.3 – 3GPP releases 15, 16 and 17 calendar ([4])

Release 15 focused on increasing throughput and interworking between 4G and 5G, and introduced the notion of URLLC (Ultra-Reliable Low-Latency Communication). URLLC consists in ensuring low packet loss and low latency communications. Indeed, several use cases (smart factories, control operations, ...) needs highly reliable communications in which the latency must be guaranteed. The objective of this thesis is to compute schemes for low latency communications. To do so, the network equipments must be able to manage the traffic by discriminating different kinds of flows and following computed scheduling to manage them. Even if current network does not yet have such capabilities, releases 16 an 17 have deepened the notion of URLLC and this topic is widely studied.

1.1.3 Current way to manage networks : Statistical Multiplexing

The constraints expressed for low latency architectures and 5G standard are hard to meet in current networks. In IP or even Ethernet networks, the traffic usually suffers of delay due to contention.

As we just mentioned, the current network nodes (routers for IP networks, switches for Ethernet networks) are not able to schedule packets. The only function of the nodes is to forward the packets to the right output port. The objective is to offer a good average quality of service for a minimal price. When a single input flow uses an output port, there is no issue. If several packets coming from several flows require the same output port at the same time, we talk about **contention**. Some packets are then put in a **contention buffer** until the port is available. The additional latency induced by contention buffers is one of the most important causes of delay. In order to avoid this situation, the bitrates of the links is dimensioned according to the use case. It is then calculated according to the

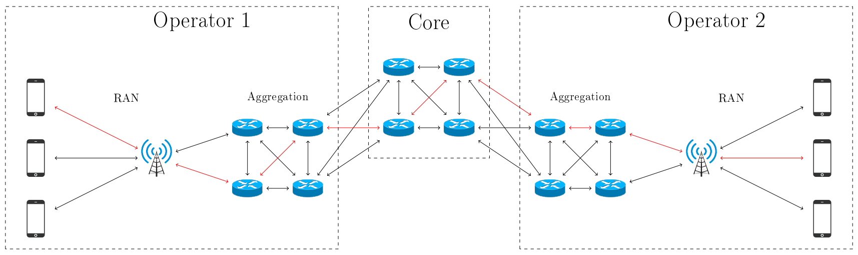

FIGURE 1.4 – An End to End communication between two mobiles.

average bitrate of the flows on the network. When there is too much packets in the buffer, the oldest packets of the buffer are **lost**.

Such an approach ensures an easy deployment and management of the networks and most of the time a good quality of service for a minimal cost for network providers, but it may induce packet losses and high latencies. This is called statistical multiplexing [5, 6].

URLLC aims to ensure good end to end latency communications. To achieve such a goal, each component of the communication network must satisfy a low latency : the radio communications, and the core network. This thesis focuses on the core network.

1.1.4 Radio Access Network

To understand the network this thesis focuses on, we now describe how a radio access network works.

Current mobile network (aka cellular network) architecture consists in a distributed radio access networks : the mobile terminals connect to a base station (BTS for Base Transceiver Station as a generic name, eNB for evolved Node B in 3GPP LTE "4G" standard or gNB for 5G) that encompasses all the sub-systems needed to realize mobile communication [7]. It mainly comprises the radio part, that furnishes the connection between the mobile terminal and the BTS, and the network part that provides control and management functions like mobility support (the main functionality being the support of handover from one BTS to another, i.e. the ability to pursue a communication when moving from the range of an antenna to another). The BTS are connected together by the Aggregation Network of the operator, itself connected to the core network, in order to ensure communications with other operators Radio Access Networks(RANs). Figure 1.4 illustrates a communication between two mobiles using a different operator.

1.1.4.1 Cloud RAN

One possible direction for next generation networks is to become centralized radio network architectures (C-RAN, for Cloud Radio Access Network) to reduce consumption costs and power at the base stations [8]. These C-RAN architectures include simplified base stations on the field. Depending on the architecture choice, it can be restricted to the radio part and the digital to analog conversion only. This can be identified by different names depending on the reference document, including **RU** for Remote Unit or **RRH** for Remote Radio Heads. The latter will be used in the thesis. The other components of the C-RAN are the processing units. One can distinguish two levels of processing units : the **DUs**, for Distributed Units that are able to ensure only a part of the computation tasks, and the **CUs** for Centralized Units that computes the most centralized tasks. In this thesis, we consider only CUs, that are also called **BBUs** for BaseBand Units, and we use this term in the thesis. The BBUs are located in the cloud.

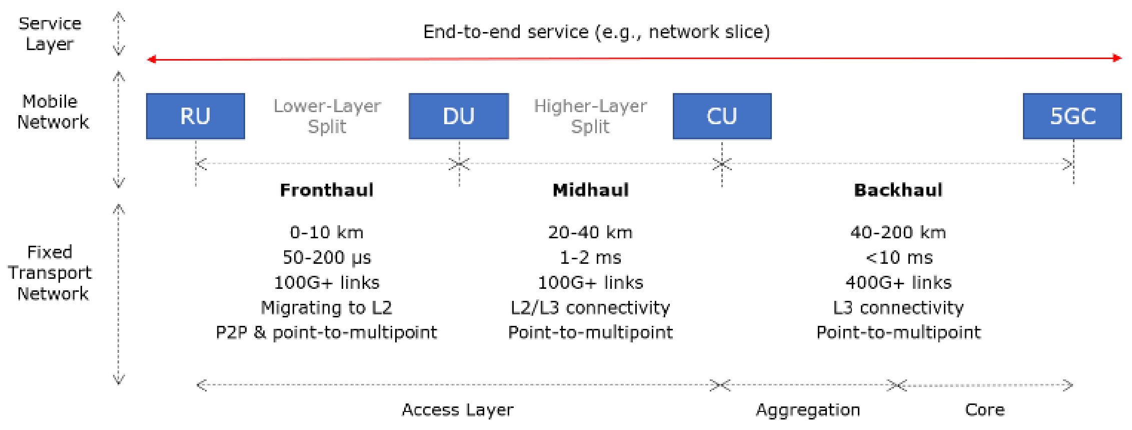

FIGURE 1.5 – Latency requirements between different part of the network.

The cloud is defined as the execution of a program in a data center (gathering the BBUs) ordered by another machine (RRHs) connected to the data center through a transparent network. The execution may be indifferently performed on virtualized machines, or bare metal, or any combination of the two. The network between RRHs and BBUs is called “Fronthaul Network”, or “Fronthaul” for short. Figure 1.6 illustrates an example of fronthaul in which several BBUs are gathered in the same datacenter.

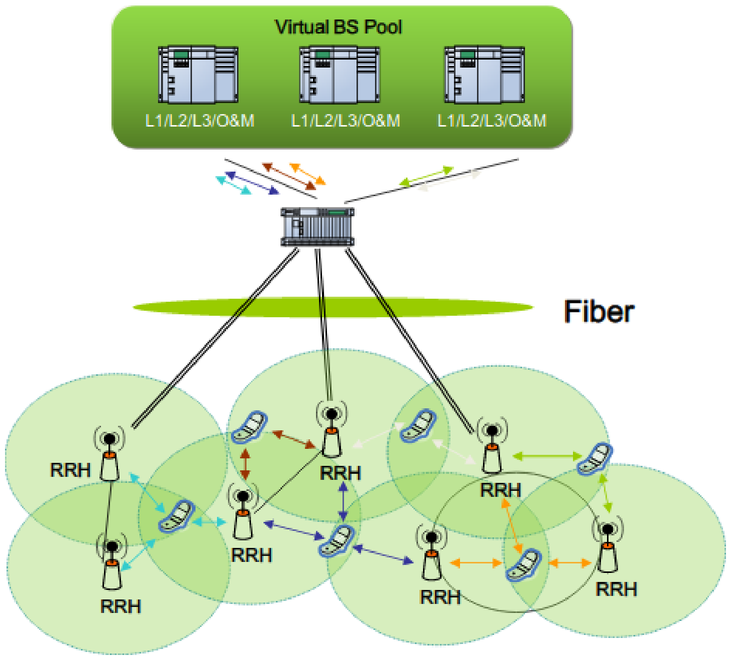

FIGURE 1.6 – An example of fronthaul network for Cloud RAN

1.1.4.2 Split

As mentioned above, in C-RAN, most of the computation tasks of the BTS must be centralized in the BBU. There are several components that can be centralized, but the more we centralize the ressources, the higher the latency constraints are. Figure 1.7 illustrates two different choices of split for the BTS. The first one, called "Full centralization" leaves only the radio functions to the RRH, while the second one, called "partial centralization", keeps the baseband processing function inside the RRH. The methods presented in this thesis allow to send an high amount of data in the network while minimizing latency. Such a result matches well with the first split, this is why we talk about BaseBand Units (BBU) here. Even if we choose to work with the most critical split, the amount of data is just a parameter of the problem presented here, and any kind of split could be managed by our algorithms.

This kind of architecture must address the problem of the latency in the transfer process between the RRHs on the field and BBUs in the cloud. Low latency is already critical for the deployment of C-RAN approach in LTE "4G" networks. The standard requires hard time constraints for functions like HARQ (Hybrid Automatic Repeat reQuest) that needs to be processed in less than $3ms$ [7]. Considering processing time into the BBU, the time

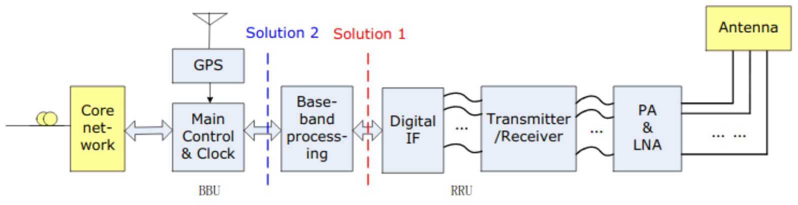

FIGURE 1.7 – Two different split for Cloud-RAN

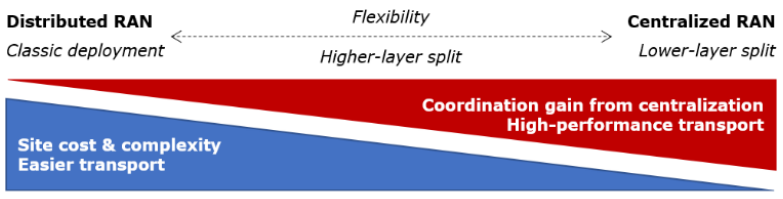

FIGURE 1.8 – The more centralized is the RAN, the hardest is the transport and the lowest is the cost of coordination for the antennas

budget over the network can be as low as $400\mu s$ for a round trip. One specificity in this C-RAN context is not only the latency constraint, but also the periodicity of the data transfer between RRH and BBU (this HARQ constraint must be enforced for each frame emitted every millisecond). Looking beyond current mobile network generation, one must have in mind that upcoming 5G standards will require to reach end-to-end expected latency as low as $1ms$ (depending on targeted services) [9]. New scheduling and new technologies have to be considered to guarantee delay constrained periodic data transfers.

1.1.5 Technical solution for low latency

Statistical multiplexing is the most common mechanism used to manage packet based networks in the last 40 years. While tools [10] ensure a latency lower than a given value for 95% of the packets, such a guarantee is not sufficient in our context in which all packets must satisfy latency constraints. Indeed, mechanisms like Express Forwarding [11] can be used to prioritize some packets over the others, but they fail to guarantee the delivery of a given packet in a given time delay when several packets compete for the same resource.

The best current solution is to rely on an almost full optical approach, where each end-point (RRH on one side, BBU on the other side) is connected through direct fiber or full optical switches [12, 13]. This architecture is very expensive and hardly scales in the case of a mobile network. As illustrative purpose, a single (one operator) mobile network in France is composed of about 10,000 base stations. This number will increase by a factor of 2 to 20 with the emergence of “small cells” which increases base station density to reach higher throughput [14, 15]. It is thus needed to find a solution to offer low latency over commoditized packet based networks.

Although 3GPP standards for 5G are not completely frozen yet, the core network is designed to use ethernet technology. Time Sensitive Networking (TSN) [16, 17] is a task group of IEEE that develops some standards for ethernet. We focus on several of those standards which allow a control of the latency. The model and the algorithms of this thesis make the hypothesis that the network components are detailed, able to collect some information and send it to a centralized entity, to differentiate several kinds of flow and to forward it at an exact date, imposed by the controller. TSN standards propose technical solution for those hypothesis, but an approach like TSN is still based on statistical laws and is limited to ensure a perfect control of the latency. TSN technologies ensure an end-to-end latency bounded for all packets, but not minimal, as we propose. The limits of TSN and the technical solution we propose are detailed in Chapter 7.

In this thesis, we work on deterministic flows. Thus, we propose algorithms to compute deterministic scheduling of the flows in the network, while minimizing the latency due to buffering. Remark that minimizing the buffering latency allows not only to meet latency constraints of applications, but also to leave additional time for others sources of latency (additional computations, longer length of fibers, etc...). When computing a scheduling, we must take into account the **periodicity** which makes the problem difficult to solve. Not only a datagram must not collide with the datagrams sent by the others BBU/RRH in the same period, but also in the other periods.

1.2 Algorithmic related works

We first describe the algorithmic problems studied in this thesis, see Chapter 2 for details. We consider a network in which the routing is fixed. Several flows share the network, sending periodically a message of the same size from a source to a target. This message represents several packets in practice, but in our model we consider them contiguous. The messages can also be buffered in the nodes of the network in order to let another message use a shared ressource. The objective is to compute the buffering time of every message in every node of the network, such that the global latency (i.e. the largest latency of a flow) is minimal. Note that the periodic aspect of C-RAN makes the problem more difficult.

Indeed, since the traffic is periodic and the network highly loaded, we must deal with contentions coming from successive periods while computing the scheduling. The problem we address is to compute periodic schedulings. This means we compute the solution on one period and the scheduling remains the same for every period. There are several variations of the problem according to the shape of the network, the synchronization of the messages in a period or the constraints on buffering.

We present in this section several families of problems and approaches, which are close to what is studied in this thesis but which mostly fail to model our problem faithfully or which are too general to derive useful algorithms.

1.2.1 Algorithmic Approaches

Wormhole Because we consider contiguous messages, we first focused on wormhole problems [18, 19]. Wormhole problems consider graphs representing interconnection networks, in which the vertices are the nodes of the network and the edges are the physical links. Long messages are sent in the network, using ressources during a long time. A *deadlock* occurs when several messages are waiting for each others to release a ressource. The main problem consists in avoiding deadlocks. The algorithmic solution proposed consists in multiplexing the physical channels in several virtual channels, and to develop routing algorithms to determine the virtual channel used by each message. The problem is very similar, but we do not have the same degrees of freedom : in our network, the routing is given. Moreover, our messages require the entire capacity of the links and we cannot use multiplexing, except in a variation of our model presented in Chapter 6. Furthermore, deadlocks can not occur in the networks we model, since the considered routings are **coherent**[20]. A routing is coherent if two routes share a single path (i.e. a sequence of contiguous links) in the network.

On a technical aspect, wormhole switches [19] are designed to read only the header of the messages before forwarding it instead of buffering the entire messages as in store-and-forward [21]. This method has a huge impact on the latency, particularly on long messages, and we go further by trying to remove all buffering in the switches.

Graph coloring for ressources allocation One of our approach consisted in representing the shared ressources of the network by a conflict graph. The vertices of the graph represent the routes of the networks and there is an edge between two vertices if their associated routes share the same ressource (i.e. an output port on a switch). The edges of the graph are labeled by the difference of the physical length of the links between the sources of the routes and the conflict point. A proper coloring of such a graph is a scheduling of the network without collisions between the messages. Several graph colorings have been introduced to model similar problems. In [22] the objective is to minimize the number of wavelength in shared links for optical networks. Flow allocations problems are studied in [23], applied to the allocation of the frequencies in radio networks. Unfortunately, they do not take into account the periodicity of the scheduling and the associated problems are already NP-complete.

Circular Coloring The only coloring taking into account periodicity is the circular coloring [24, 25]. As an example, circular coloring can model a road intersection as a graph in which the vertices are the flows and there is and edge between two vertices if the flow must not overlap. The problem is to find a proper vertex coloring of the graph. Such an approach is close to our problem of coloring conflict graphs. However, the models presented do not consider the weight on the arcs, and cannot easily capture general graphs. Also, because the problem is already NP-hard , this approach did not catch our attention. Circular arc coloring [26] also manages jobs that can be related to periodic messages. The authors consider a clock and several arcs around this clock, representing the time needed by a job. The objective is to find the minimal number of men needed to complete the job. If two arcs overlap, the same man cannot be affected to both jobs. Nevertheless, this problem has been shown NP-hard [27] and the model is really far from ours.

1.2.2 Scheduling approaches

Train Scheduling The train timetabling problem [28] and its restriction, the periodic event scheduling problem [29] are generalizations of the problem we study. Indeed, they take the period as input and can express the fact that two trains (like two messages) should not cross. However, they are much more general : the trains can vary in size, speed, the network can be more complex than a single track and there are precedence constraints. Hence, the numerous variants of train scheduling problems are very hard to solve (and always NP-hard). Most of the research done [28] is devising practical algorithms using branch and bound, mixed integer programming, genetic algorithms. . .

Linear Programming for Latency Constrained Network We define in this thesis a simple network topology called the *star shaped network*. In star shaped networks, all flows go through the same link, and there is only two relevant contention points (one in the way to the BBUs, and one in the way back to the RRHs).

Variation on the problem of scheduling periodic messages for Cloud-RAN have been investigated in [30, 31, 32, 33]. In these papers, authors study the practical problem of scheduling a few number of flows in a star shaped network. Given a network in which the routing is set, the objective is to schedule several periodic flows with a critical latency and several best-effort flows, as we do in Chapter 6. To do so, the authors use new technical standards (TSN, detailed in Chapter 7) that allow to prioritize flows, and linear programming in order to compute a schedule between the critical flows. In the same spirit, the use of an SMT solver rather than linear programming is proposed in [34]. The flows described in these papers are different from ours. While we consider that a single long message is sent by a source every period, the authors propose flows in which several little packets are sent. The scheduling are computed on multiple periods while we compute a scheduling on one period, which can be repeated periodically. Furthermore, the experiments are made on small topologies, because these approaches does not scale neither with the number of routes nor the number of conflict points on each route, while we propose polynomial time algorithms that give satisfying solution for every kind of topology. However, this kind of approach, as explained in [31], can be used as a standardization tool to verify the viability of solutions computed by faster algorithms.

A polynomial algorithm for single processor scheduling The problem we address in this thesis is similar to classical single processor scheduling problems, defined as follows. Given a set of jobs, a release time and a deadline for each job, find a scheduling to minimize the global completion time (i.e. the time at which all jobs have been processed). The approach we adopt in Chapter 4 to solve our problem on a star routed network, is a two stage approach. The second stage, when forgetting about periodicity, is similar to a scheduling problem with a single processor with the goal of minimizing the completion time (makespan). The algorithm described in [35], solves this problem in polynomial time when all tasks have the same processing time. We use it as a building block of several of our algorithms in Chapter 4 and Chapter 5, where we adapt it to the case of periodic jobs. When the jobs have different processing times, the problem is NP-hard [36].

Two flow shop scheduling On the star routed network studied in Chapter 3, the problem of scheduling messages for Cloud-RAN without buffers nor periodicity is similar to a two flow-shop scheduling problem. In two flow-shop scheduling, the objective is to schedule the jobs on two processors, each job must be first processed by the first processor and can then be processed by the second one after a delay which depends on the job. This two-flow shop problem is NP-hard [37], and while it can be reduced to the synchronized version of our problem, presented in 5, the correspondence is less clear with the unsynchronized version of Chapter 3. Moreover, our problem adds the constraint of periodicity, hence no algorithm for two flow shop scheduling can be used as is.

The problem of scheduling tasks on multiple processors periodically does not seem related to our problem. In [38] the problem is the following : Given a set of tasks, with for each of them a duration and a period, find a periodic scheduling minimizing the number of processors on which the periodic tasks are scheduled. The model is different from ours because we consider messages of same size and period, and we want to schedule all messages on the same ressource.

In general, the problem of cyclic scheduling [39, 40] have been extensively studied. The problem has many variants, considering or not precedence constraint, different message sizes, or different periodicity for each message, etc... Most of these problems have been shown to be NP-hard. In all these problems, the objective remains the same : minimizing the period. The main difference, which makes methods for cyclic scheduling quite different from our approach, is that the period is an input of our problem, which cannot be modified. While cyclic scheduling tries to optimize the throughput by changing the period, we try to optimize the latency for a fixed period.

Conclusion

One of the trend of 5G and networks in general is to be able to ensure end to end communication with a low latency. To do so, it is important to reduce the latency sources in each part of the network. In this thesis, we focus on the packet switched network connecting the radio antennas to the operator's core network. In packet switched networks, the major source of latency is the buffering latency due to contention. Current approach of network management consists in multiplexing the flows on a network dimensioned following statistical laws, and this may induce high contention buffering time. In order to reduce the contention, several technical solutions are currently in development to control critical traffics more precisely than statistical multiplexing. With those solutions, the switches follows all over the time a precise scheduling of each flow it must forward at each date. In this thesis, we study the problem of computing this scheduling. Knowing the topology of the network, and the different flows it supports, our objective is to compute the scheduling of the traffic for every node of the network, while minimizing the latency of the flows. Because we consider applications sending all over the time a datagram periodically, the scheduling must be periodic. Scheduling problems for networks have been extensively studied in wormhole routing, train scheduling or frequency allocation, but none of these studies deal with the periodicity of the flows we want to schedule, which makes the problem innovative and difficult. Circular scheduling or coloring problems seems related to our problem but the model are too far from ours, and the problem are not the same. Single processor scheduling, with makespan constraint can be adapted for periodicity to build some of our algorithms.

Chapitre 2

Model, Problems

The model and the proofs presented in this chapter has been introduced in [41] and extended in a long version of the paper in [42].

2.1 Model

Let $[n]$ denote the interval of n integers $\{0,\dots,n-1\}$.

2.1.1 Routes and Contention Points

We study a communication network with pairs of source (s) target (t) nodes between which messages are sent periodically. The routing between each pair of such nodes is given : a **route** is a sequence of vertices $(s, c_1, \dots, c_l, t)$. A vertex appears only once in a route. Each vertex c_i corresponds to a contention point, that is the beginning of a link of the communication network shared by several routes. Hence, all vertex appears in several routes, except the first (s) and last (t) vertex of a route, which are exclusive to the route and represent the source and the target of the message. When modeling a C-RAN network, the first vertex represents the RRH sending of the message by the RRH and the last vertex represents the the same RRH that receives the answer.

The set of routes is denoted by $\mathcal{R}$. A route is interpreted as a directed path in a directed multigraph constituted of all routes, where the sets of arcs of the routes are disjoint. The routes contain no loop nor cycle, since all vertices of a route are different. Thus, the directed multigraph is acyclic. An arc in the multigraph may represent several physical links or nodes of the modeled network, which do not induce contention points.

Each arc (u,v) of a route r is labeled by an integer weight $\omega(r,u)$. It represents the time elapsed between the sending of the message of the route r in u and its reception in v. The **weight of a vertex** u_i in a route $r = (u_0,\dots,u_l)$ is defined by $\lambda(r,u_i) = \sum\limits_{0 \leq j < i} \omega(r,u_j)$. It

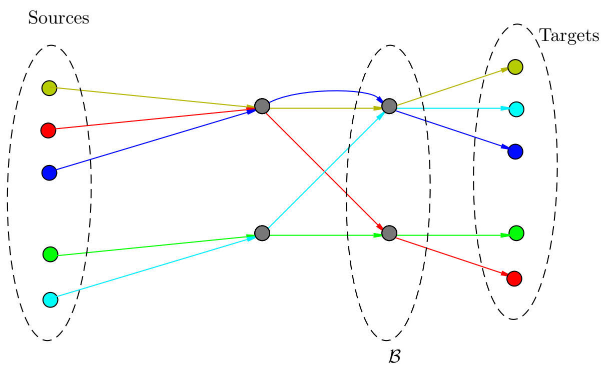

FIGURE 2.1 – A routed network, each route is represented by a colored path

is the time needed by a message to go from the first vertex of the route to u_i. The **length** of the route r is defined by $\lambda(r) = \lambda(r,u_l)$.

On each route, we can buffer the message only in the BBU. Since the BBU does not correspond to a contention point, we identify the BBU with the next contention point in the route. The set of these contention points with possible buffering is denoted by $\mathcal{B}$. Each route have thus only one vertex in $\mathcal{B}$. A **routed network**, which models the telecommunication network, is a triple $N = (\mathcal{R}, \mathcal{B}, \omega)$, see Figure 2.1 for an example.

2.1.2 Dynamic of Datagrams Transmissions

In this thesis, we consider a discretized time. The unit of time is called a **tic**. This is the time needed to send an atomic data in a link of the network. We assume that the speed of the links is the same over all the network. We are developing a prototype of this work based on ethernet base-X [43], see chapter 7, using standard values for the parameters of the network : the size of an atomic data is 64 bits, the speed of the links is 10Gbps, and the length of a tic is thus about 6.4 nanoseconds.

In the process we study, a message, called a **datagram**, is sent on each route from the source node. The **size** of a datagram is an integer, denoted by τ, it is the number of tics needed by a node to emit the full datagram through a link. In this thesis, we assume that τ is the same for all routes. It is justified by our application to C-RAN, where all source nodes are RRHs sending the same type of message. There is no fragmentation : Once a datagram has been emitted, it cannot be fragmented during its travel in the network.

Let $r = (s, \dots ,t)$ be a route. In order to avoid contention, it is possible to buffer datagrams in the contention points in $\mathcal{B}$. An **assignment** A of a routed network $N = (\mathcal{R}, \mathcal{B}, \omega)$ is a function which associates to each route $r \in \mathcal{R}$, the pair of integers $A(r) =$

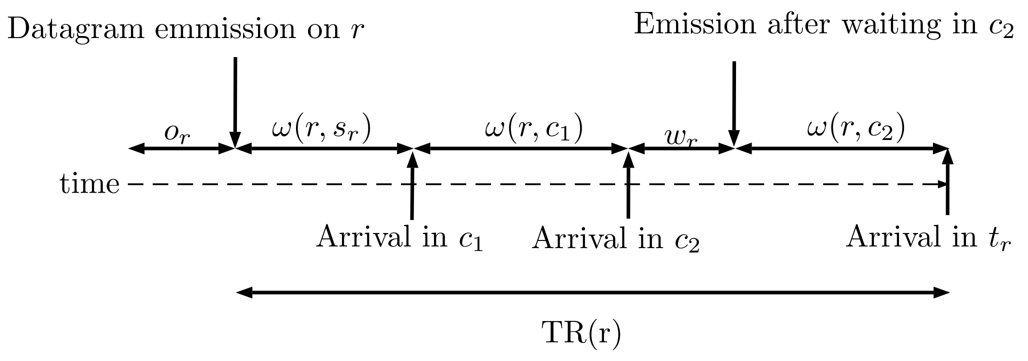

FIGURE 2.2 – Timeline of a datagram during its travel on a route $r = (s_r,c_1,c_2,t_r)$, with $c_2 \in \mathcal{B}$

(o_r,w_r). The value o_r is the **offset**, the time at which the datagram is available in the first vertex of r. The value w_r is the **waiting time** : the datagram is buffered for w_r tics in $u_j \in \mathcal{B} \cap r$, the vertex representing the BBU. The **arrival time** of a datagram in the vertex u_i of r, is the first time at which the datagram sent on r reaches u_i, and is defined by $t(r,u_i) = \lambda(r,u_i) + o_r$ if $i < j$ and $t(r,u_i) = \lambda(r,u_i) + o_r + w_r$ otherwise.

Let u_l be the last vertex of the route r, the **transmission time** of the datagram on r is denoted by $TR(r,A)$ and is equal to $\lambda(r) + w_r$ or equivalently $t(r,u_l) - o_r$. This is the total time taken by the process we study : the sending of the datagram from the RRH to the BBU and the return of the answer back to the RRH. We can decompose this time into $\lambda(r)$, the *physical latency* of the process and w_r, the *logical latency*. We define the **transmission time** of an assignment A as the worst transmission time of a route : $TR(A) = \max_{r \in \mathcal{R}} TR(r,A)$. Figure 2.2 represents the different events happening during the lifetime of a datagram sent on a route r.

2.1.3 Periodic Emission of Datagrams

In the previous section, we have explained how *one* datagram follows its route. However, the process we model in this thesis is *periodic* : for each period of P tics, a datagram is sent, from each source node in the network, at its offset. The process is assumed to be infinite, since it must work for an arbitrary number of periods. For a given route, we use the same offset and waiting time in all periods, for simplicity of implementation in real networks and to make our problem more tractable from a theoretical perspective. Hence, at the same time of two different periods, all datagrams are at the same position in the network : the assignments built are themselves periodic of period P. Thus, we only need to consider the behavior of the datagrams on each node of the network during a single period, and to apply the same pattern to every subsequent period. Using a different offset for each route corresponds to sending their datagram at a different time in the period. This matches our hypothesis that the emissions of the RRHs need not to be synchronized but they share a common global time, useful for coordination of their emission.

Let A be an assignment of a routed network $N = (\mathcal{R}, \mathcal{B}, \omega)$. Let us denote by $[r,u]_{P,\tau}$, the set of tics used by a datagram on the route r at vertex u in a period P, that is $[r,u]_{P,\tau} = \{t(r,u) + i \mod P \mid 0 \leq i < \tau\}$. This set of tics depends on A, but A is omitted in the notation, since it is always clear from the context. Let us consider two routes r_1 and r_2, they have a **collision** at the contention point u if and only if $[r_1,u]_{P,\tau} \cap [r_2,u]_{P,\tau} \neq \emptyset$. The assignment A is said to be **valid** if, for all contention points u and routes r_1 and r_2 containing u, r_1 *and* r_2 *have no collision* at u. The validity of an assignment depends on P the period and τ the size of the datagrams, thus we say that A is a valid (P,τ)**-assignment**. When P and τ are clear from the context, we denote $[r,u]_{P,\tau}$ by $[r,u]$ and say that A is a valid assignment.

The following Figure 2.3 illustrates two valid periodic assignment for different values of P and τ, and the criticality of waiting time in latency reduction. The three routes are depicted by three different colors. If we let $P = 2$ and $\tau = 1$, then there is a (2,1)-periodic valid assignment with waiting times zero by taking 0 as offset for each route. However, for the same routed network but $P = 5$ and $\tau = 2$, there is no solution to the problem with waiting times zero. If we allow 1 tic of waiting time for one route, we can build the valid assignment ((0,0),(2,1),(0,0)).

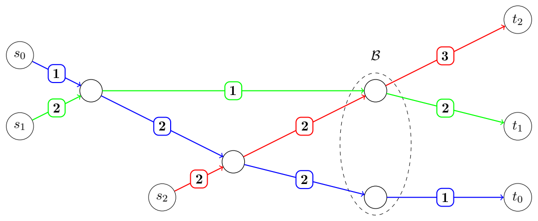

FIGURE 2.3 – A routed network with ((0,0),(0,0),(0,0)) as a (2,1)-periodic valid assignment and ((0,0),(2,1),(0,0)) as a (5,2)-periodic valid assignment

2.1.4 Periodic Assignment for Low Latency

The period P, as well as the size of a datagram τ are fixed in our C-RAN settings, but not the buffering policy. Hence, the aim of this section is to find a valid assignment which minimizes the worst latency of the transmissions over the network, that is $TR(A)$. We denote by MINTRA the problem of finding the minimal value of $TR(A)$, for a given period, datagram size and routed network. For simpler hardness proofs and easier reductions, we rather study the decision version of MINTRA, that we call PALL for **P**eriodic **A**ssignment for **L**ow **L**atency. Each route must respect a time limit called a *deadline*. These limits are encoded in a deadline function d, which maps to each route r an integer such that $TR(r,A)$ must be less than $d(r)$. We define the **margin** of a route r in a routed network N with deadline function d as $d(r) - \lambda(r)$. The margin of a route is the maximum possible waiting time of the route.

Periodic Assignment for Low Latency

Input : A routed network N, the integers P, τ and a deadline function d.

Question : Does there exist a (P,τ)-periodic assignment of N such that for all $r \in \mathcal{R}$, $TR(r,A) \leq d(r)$?

In the next section, this problem is proved to be NP-hard. In Chapter 4, we propose heuristics solving the search version of PALL (computing a valid assignment), also denoted by PALL for simplicity. In the definition of PALL, we have chosen to bound the transmission time of each route, in particular we can control the worst case latency. It is justified by our C-RAN application with hard constraints on the latency.

We say that an assignment is **bufferless** when the waiting time of all routes are zero. The assignment can then be seen as a function from the routes to the integers (the value of the offset, the waiting time is omitted). We consider a restricted version of PALL, requiring to find a bufferless assignment and studied in Chapter 3. This is equivalent to using the deadline function $d(r) = \lambda(r)$, that is the transmission time must be equal to the size of the route, which implies $w_r = 0$ for all $r \in \mathcal{R}$. This problem is called **P**eriodic **A**ssignment for **Z**ero **L**atency and is denoted by PAZL. Studying PAZL is simpler : in an instance, there is no need to precise $\mathcal{B}$ in the routed network nor the deadline function and a solution is just an offset for each route. Moreover, a solution to PAZL is more efficient in real telecommunication networks, since we do not need to deal with any buffering at all. A switch taking full advantage of the absence of buffer is presented in Chapter 7.

An unusual property of assignments is that given a routed network and a deadline, we may have a (P,τ)-periodic assignment but no (P',τ)-periodic assignment with $P' > P$: the existence of an assignment is not monotone with regard to P.

Proposition 1. *For any odd P, there is a routed network with a $(2,1)$-periodic bufferless* *assignment but no* $(P,1)$*-periodic bufferless assignment.*

Démonstration. Let us build N, a generalization of the routed network given in Figure 2.3. Let n be an integer, the vertices of the routes are the $v_{i,j}$, v_i^1 and v_i^2, with $0 \leq i < j < n$. There are n routes denoted by r_i, for $i \in [n]$. The route r_i is equal to $(v_i^1, v_{i,1}, \dots, v_{i,n-1}, v_i^2)$. The weights of the arcs are set so that $\lambda(r_i, v_{i,j}) - \lambda(r_j, v_{i,j}) = P$, where P is an odd number smaller than n. It is always possible by choosing appropriate values for $\omega(r_i, v_{i,j-1})$ and $\omega(r_j, v_{i-1,j})$. In such a graph, there is no (P,τ)-periodic assignment with zero waiting time, since the problem reduces to finding a P-coloring in a complete graph with $n > P$ vertices, the colors being the offsets of the routes.

If we consider a period of 2, for all $i \neq j$, $\lambda(r_i, v_{i,j}) - \lambda(r_j, v_{i,j}) \mod 2 = 1$, hence two datagrams of same offset and size 1 do not have a collision at $v_{i,j}$. Therefore, the bufferless assignment defined by $A(r_i) = 0$ for all $i \in [n]$ is a valid $(2,1)$-periodic assignment of N. □

Let us introduce a few parameters quantifying the complexity of a routed network. The **contention depth** of a routed network is the number of vertices of the longest route of the network minus two. It is the number of contention points on the route with the most vertices on the network, since the first and the last vertex are private to the route. The **width** of a vertex is the number of routes which contains it. By definition, the first and last vertex of a route are of width 1, while all other vertices are of width at least 2 (otherwise they can be removed). The **contention width** of a routed network is the maximal width of its vertices. Remark that a (P,τ)-periodic assignment of a routed network must satisfy that P/τ is larger or equal to the contention width. Now, let us fix P and τ, for a given vertex of contention width c, we define its **load** as $c\tau/P$. It represents the proportion of the period used by datagrams at this contention point. The load of the routed network is the maximum of the loads of its vertices. A routed network must have a load less or equal to one to admit a valid assignment.

2.1.5 The Star Routed Network

In this section, we define a family of simple routed networks modeling a Multipoint-to-Multipoint fronthaul (see figure 2.4), which has been designed for C-RAN [13]. Let $N = (\mathcal{R}, \mathcal{B}, \omega)$ be a routed network, we say it is a **star routed network** if and only if the routes are $\{r_0, \dots, r_{n-1}\}$, r_i is (s_i, c_1, c_2, t_i) and $\mathcal{B} = \{c_2\}$ (datagrams can wait in c_2). Star routed networks have contention depth two but a maximal contention width of n. The load