# Routing and slot allocation in 5G hard slicing

Nicolas Huin, Jérémie Leguay, Sébastien Martin, Paolo Medagliani

# Routing and Slot Allocation in 5G Hard Slicing

Nicolas Huin Jérémie Leguay Sébastien Martin Paolo Medagliani

February 22, 2023

**Abstract**

Current network slicing solutions suffer from poor inter-slice isolation, as the performance of one slice can be influenced by the traffic in other slices. New technologies such as Flex Ethernet can offer hard isolation via dedicated resources at the physical and MAC layers. However, to create cost-efficient hard slices in large 5G access networks, a "routing and slot allocation" must be solved quickly. While the underlying network design problem is not new, two extra constraints need to be considered: a specific order in slot activations and a bandwidth allocation policy with statistical multiplexing.

We propose a compact and extended formulation to derive FlexE-CG, an algorithm based on column-generation to solve large instances. We reinforce the extended formulation to improve the lower bound by deriving valid inequalities, and we provide necessary and sufficient conditions under which the inequalities are facet-defining. We show that these inequalities improve the lower bound by more than 20% on various IP-Radio Access Networks (RAN). We also show that FlexE-CG can provide solutions within an optimality gap of 10% in a few minutes.

## 1 Introduction

The deployment of 5G mobile networks is paving the road to custom network services with a rapid increase of network capacity and the advent of network automation [20]. It is now possible to envision the automatic decomposition of the physical network into several virtual networks to serve a wide range of applications like factory automation, connected vehicles, and smart grids. Each virtual network, also called a *slice*, has dedicated resources and is operated by a different player, often referred to as *tenant* (e.g., a customer, an application). A virtual link, also called a *service*, between physical nodes can be realized as a multi-hop path with reserved resources on all physical links constituting the path. The partitioning of network resources between slices aims at guaranteeing that tenants' requirements, in terms of QoS, are met in all slices. Depending on the slicing technology used, the traffic in one slice can potentially interfere with the traffic in other slices. Two levels of isolation exist: *soft* and *hard* slicing.

In soft slicing [17, 14], traditional QoS and routing mechanisms prioritize traffic and allocate bandwidth, but performances are still statistical; resources are pledged to slices, but the traffic is multiplexed in a stochastic queuing system. When one slice overloads a physical link, it induces an extra latency for all other slices sharing that link: it is impossible to strictly ensure that the traffic in one slice will not impact other slices. Both IP (Internet Protocol) and MPLS VPNs (MultiProtocol Label Switching Virtual Private Networks) are examples of soft isolated virtual networks because the network delivers the traffic only to the required VPN endpoints. The main difference between IP and MPLS concerns how packets are routed to the destination, providing QoS satisfaction.

On the contrary, in hard slicing [23], each slice has dedicated resources at both physical and MAC layers, which leads to deterministic performance; misbehavior of one slice cannot influence other slices. Extensions of VPN technologies for hard slicing are under discussion at IETF and known as *Enhanced Virtual Private Networks (VPN+)* [18], a technology to support the needs of new applications (e.g., low latency, bounded jitter), by utilizing an approach that is based on the VPN and Traffic Engineering (TE) technologies.

Flex Ethernet (FlexE) is a promising technology that provides hard isolation in IP networks by reserving dedicated resources at the physical and the MAC layers in a Time-Division Multiplexing Access (TDMA)

fashion. The capacity of physical ports is allocated to each slice in the form of capacity slots, i.e., multiples of a fundamental bandwidth unit, usually expressed in Gigabits. The slots are of different sizes and need to be activated in a given order. Once a slot is allocated to a slice, it cannot be shared with another one. In this way, FlexE can provide a performance guarantee for the customer, especially in transport networks, where this technology is expected to be deployed. When a slice is created, FlexE slots must be reserved on physical links and user traffic must be steered through these slots.

A network controller is typically taking routing and slot allocation decisions with the goal of minimizing reserved resources globally. A common way to avoid over-provisioning in IP-RAN (IP Radio Access Networks) is to use statistical multiplexing across the different levels of the network topology (i.e., access, aggregation, core) as not all services are active at the same time. In this way, the network operator can still satisfy bandwidth requirements from a customer while minimizing the required resources assigned to him. In this context, we present the *Routing and Slot Allocation (RSA)* problem for hard slicing with FlexE in 5G networks where the goal is to minimize the cost of bandwidth reservations for a single slice, under the constraint that all services of the slice are accepted. The problem considers the statistical multiplexing policy and the activation order of slots.

Since we consider only one slice, the RSA problem is similar to the *capacitated network design problem with general step cost functions* [22] or the *energy-aware routing with discrete link rates problem* [2]. While neither work studies network slicing, the structure of their respective studied problem relies on the discrete allocation of resource. However, key differences prevent us from using their solutions. First, Gabrel et al. [22] consider splittable flows, unlike our problem where each service must be routed on a unique path. Second, even though Awad et al. [2] consider unsplittable flows, we cannot apply their solution as in this paper the authors do not consider statistical multiplexing, which is, de-facto, a standard and widely adopted solution in IP-RAN networks (see Section 2.4).

Only a few works have considered statistical multiplexing for slicing or network design. Belotti et al. [6] and Capone et al. [11] considered the case of multi-layer networks where MPLS is considered on top of transport networks (e.g., SDH or WDM). Statistical multiplexing is used at the MPLS level in complement to traffic grooming [27] at the optical layer to better exploit the large capacity available on each wavelength. Packet-switched networks can indeed aggregate traffic using a pool of shared resources, and they require much less capacity than what would be needed from an equivalent aggregation at the optical level with grooming. The authors use Lagrangian relaxation to derive upper and lower bounds to the problem and to evaluate greedy and local search heuristics.

When the traffic demand can be precisely known and statistical multiplexing is not considered, several approaches have been proposed to solve the capacitated network design problem using, for instance, decomposition methods and cutting planes [15, 21, 30]. None of these inequalities are suitable for our problem as different assumptions are made on the slot allocation policy (e.g., uniform size, uniqueness) and statistical multiplexing is not considered. Section 4.2 provide a deeper analysis of these cutting planes compared to the ones we propose.

As in some cases the traffic may not be predictable, robust optimization methods under uncertainty sets have been proposed. The robust network design problem where a linear reservation cost is minimized was proved to be co-NP hard [12]. This problem is also referred to as *dynamic routing* in the literature since the network is optimized such that any realization of the traffic matrix in the uncertainty set has its own routing. As opposed to dynamic routing, *static routing* or *oblivious routing* was introduced in [7, 1, 4] to choose a fixed routing for all services. In this case, polynomial-time algorithms to compute optimal static routing (with respect to either congestion or linear reservation cost) have been proposed [7, 8, 1, 4] based on either duality or cutting-plane algorithms. To further improve solutions of static routing and overcome complexity issues related to dynamic routing, a number of restrictions on routing have been considered to design polynomial-time algorithms (see [9, 29] for a complete survey). We leave the robust case for future work and focus on the deterministic case, where a single and known matrix defines the traffic load of all services. In particular, for each of them, we must provide an end-to-end path respecting the requested QoS.

Our paper makes the following contributions. First, we formulate the RSA problem. Then, we present an efficient heuristic based on column generation [16] and randomized rounding to quickly approximate the

optimal solution. Thanks to a polyhedral analysis, we reinforce the original model with cutting planes to improve the lower bound given by the optimal solution of the relaxed model. We propose three families of inequalities: cover inequalities, edge-cuts inequalities, and 1-slot inequalities. We provide necessary and sufficient conditions under which the edge-cut and 1-slot inequalitie are facet-defining and sufficient conditions under which the cover inequalities are facet-defining. Through numerical results on large-scale and realistic IP-RAN instances, we confirm that these inequalities improve the lower bound and we show that the column-generation-based heuristic provides a high-quality solution to the problem. To the best of our knowledge, we are the first to propose a column-generation-based heuristic and cutting planes to efficiently solve the RSA problem with statistical multiplexing.

The structure of the paper is the following. We explain *hard slicing* from a technological point of view and present the RSA problem in Section 2. We then formally propose a compact formulation of the problem and give an extended formulation in Section 3 that we use in a column generation procedure. Furthermore, we derive valid inequalities and perform a polyhedral analysis in Section 4 to improve the lower bound. In Section 6, we show on 5G scenarios, using an IP-RAN network, the improvement of the lower bound and the performance of the column generation algorithm when using the different inequalities to reinforce the master problem. Finally, we conclude this paper in Section 7.

# 2 Routing and Slot allocation in IP-RAN networks

This section provides background information on Flex Ethernet and explains how bandwidth has to be allocated for hard slicing in IP-RAN networks with statistical multiplexing. At the end of the section, we introduce the RSA problem that a slicing controller needs to solve.

## 2.1 Flex Ethernet

The Optical Internetworking Forum (OIF) has designed a new technology, referred to as Flex Ethernet (FlexE) [28], which allows network operators to physically separate the resources of different tenants by adding specific FlexE shims at both head-ends of transmission links. While FlexE has been conceived for interfacing IP and transport networks, it allows the support of previously unavailable Ethernet rates and can be used to provide hard isolation between IP network slices.

As shown in Figure 1 (a), several FlexE clients, corresponding to different users (i.e., tenants in the hard slicing case), are handled by a FlexE shim, a sub-layer that is added between the PHY and the MAC layers of a standard IEEE 802.3 stack. The FlexE shim is responsible for mapping and de-mapping the FlexE clients into the bounded physical FlexE groups, i.e., a set of $n$ clients mapped into $m$ 100 Gbps physical interfaces and de-mapped after the data is received. The operation of multiplexing is depicted in Figure 1 (b). Data from the clients are decomposed into smaller 66 B frames and sent over the link using a Time Division Multiplexing (TDM) mechanism. The associated TDM frame is divided into 20 slots, whose size is 5 Gbps each. To let the receiving FlexE shim demultiplex data received on the link, the two FlexE devices share a calendar. In this way, the receiving device becomes aware of the slots used by the transmitting device for each FlexE client (i.e., tenant in the hard slicing case).

Existing VPN solutions can already provide *soft* isolation between tenants, i.e., the different tenants are logically separated. However, congestion inside a slice may impact all the other slices as they share resources at the MAC layer. FlexE, instead, ensures *hard* isolation between tenants via a TDM frame at the MAC/PHY layer and sub-interfaces with dedicated resources at the MAC layer. Each FlexE sub-interface is mapped into a dedicated MAC block to avoid delay and bandwidth interference with other slices. The FlexE shim then maps the FlexE client (i.e., hard slicing tenant) with the corresponding physical resources according to the information in the calendar. And each FlexE sub-interface implements traditional QoS mechanisms to handle requirements of the flows of the same FlexE client (i.e., inside the hard isolated slice). The benefits of hard slicing for isolation with FlexE have been previously demonstrated [24] in the case some of the slices become overloaded.

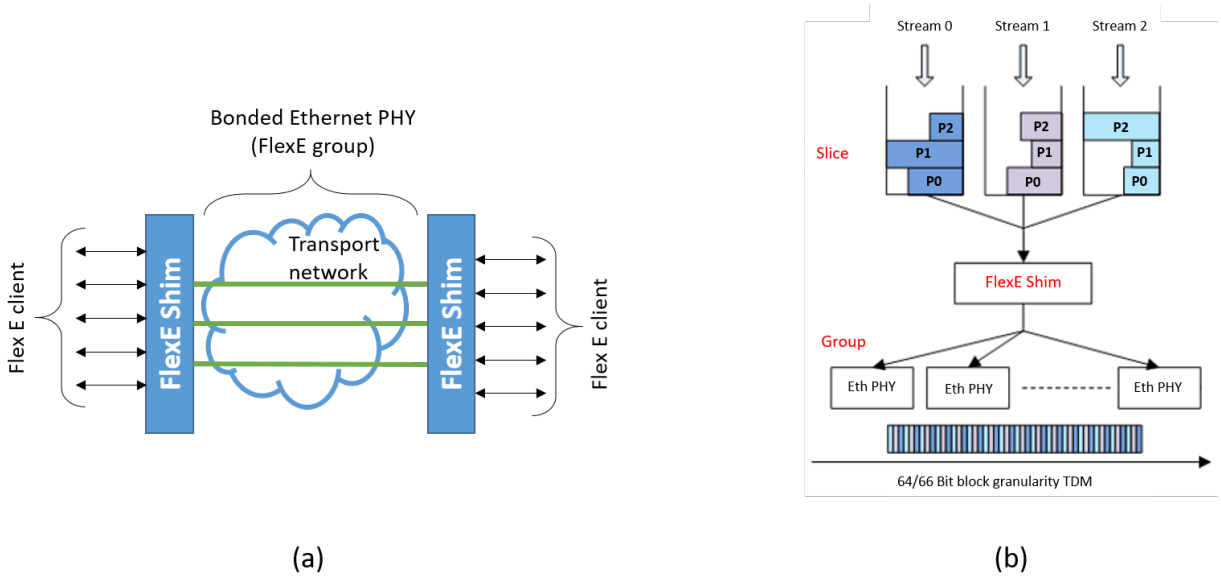


Figure 1: (a) General structure of a FlexE network and (b) TDM division of data transmissions.

Extension of VPN technologies for hard slicing are under discussion at IETF and known as *Enhanced Virtual Private Networks (VPN+)* [18]. In this proposal, a set of dedicated underlay resources, such as FlexE sub-interfaces, is advertised to the network layer. These underlay resources can be exploited using MPLS-TE [3] or Segment Routing [19]. In the case of Segment Routing, the source node explicitly states, via a Segment Routing (SR) label stack, in which sub-interface each packet must be transmitted at each link.

## 2.2 Bandwidth allocation policy

The TDM frame and the shared calendar between two adjacent nodes impact the bandwidth granularity that can be allocated to each FlexE client. In the latest version of the standard, i.e., the 2.0 [28], the finest granularity of each TDM frame is 5 Gbps. In Huawei's implementation of the standard FlexE, the first slot used by each sub-interface can present a finer granularity, i.e., it can be further subdivided into 5 slots of 1 Gbps each.

As a slot cannot be shared between different clients (i.e., slices) to ensure hard isolation, the bandwidth for each FlexE client (i.e., each slice) is reserved according to the following rules:

1. if a new slot needs to be reserved, its size must be larger than the demand size;
2. when reserving a new slot, first use the 1 Gbps slots available and, if none is left, use the 5 Gbps slots;
3. the portion of bandwidth already reserved for a slice can be used by any services of the slice.

An illustrative example of how bandwidth is reserved according to the FlexE technology is shown in Figure 2. Over a 10 Gbps link using FlexE, two services belonging to the same slice must be routed. The size of Service 1 (in red) is 7 Gbps, while the one of Service 2 (in blue) is 3 Gbps. Assuming that Service 1 arrives in the network, it is necessary to reserve bandwidth for it in the link. As it requires 7 Gbps, the first five 1-Gbps slots plus the 5-Gbps slot are used. This means that 10 Gbps are reserved to the slice including Service 1. More precisely, Service 1 will only use 7 Gbps (marked with solid color) and that 3 Gbps are allocated but not used by any service of the same slice (marked with diagonal lines). When Service 2 arrives into the network, it can use the 3 Gbps available without any extra-reservation, fully occupying the capacity reserved for the slice.

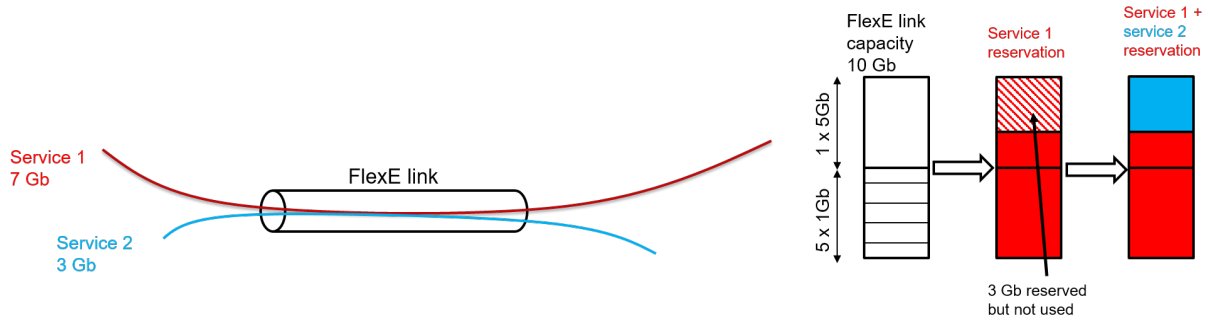


Figure 2: Example of FlexE bandwidth allocation policy on a 10 Gbps link with two services (Service 1 in red requires 7 Gbps, while Service 2 in blue requires 3 Gbps).

## 2.3 IP-RAN topology

According to the 3GPP standard, a telecommunication network can be divided into Access Network (AN), Transport Network (TN), and Core Network (CN). FlexE is expected to be used for traffic isolation and performance guarantee in the Transport Network, often implemented as an IP-RAN: a packet (IP)-based network designed to implement fixed-mobile convergence (FMC). The IP-RAN is therefore introduced to support diversified services, such as mobile, leased line, and fixed network services using the same network infrastructure.

The IP-RAN network, as shown in Figure 3, is divided into three layers: the access, the aggregation, and the core network.

The network topology has a hierarchical structure: there is only one core layer, to which several aggregation layers are connected, being themselves connected to several access layers. Each layer corresponds to one or more domains, i.e., logical subnets composed by routers and switches, to ease management and ensure scalability.

Base Stations (BS) are connected to the nodes in the access layer, referred to as Cell Site Gateway (CSG). Each access layer is connected to the aggregation layer via a pair of Aggregation Site/Service Gateway (ASG) nodes. Each aggregation layer is connected to the core layer via a pair of Aggregation Autonomous System Border Routers (A-ASBR), associated with a pair of Core System Border Routers (C-ASBR) via direct links. These latter are normally connected to each other with a full mesh. Finally, the core network is connected to the Evolved Packet Core (EPC), which corresponds to the core network described above.

## 2.4 Statistical multiplexing

For the sake of simplicity, the planning of resources is carried out in the worst-case scenario where all the services, also referred to as *demands* in the rest of the paper, are active at the same time. However, in large-scale and realistic network scenarios, this assumption may yield to a costly over-provisioning of bandwidth as in reality, some demands are only present for a limited amount of time every day. For instance, flows from one base station to another related to data or communication services are expected to be sporadic. For this reason, instead of reserving the full capacity over the links, it is possible to *statistically multiplex* the bandwidth of services of the same type — i.e., assign only a portion of the bandwidth using statistical information on the coexistence of different services — and reduce the overall amount of bandwidth necessary to route all the services.

Statistical multiplexing is particularly useful in the aggregation and core layers, where capacity is higher and a larger number of services are using this part of the network. Conversely, we can assume that no multiplexing is needed in the access layer. For this reason, in the network planning of bandwidth reservations, we consider the possibility to scale down the reserved bandwidth by a multiplicative factor, as shown in Figure 4, referred to as *Convergence Ratio* (CR), which depends on the considered layer. For the sake of

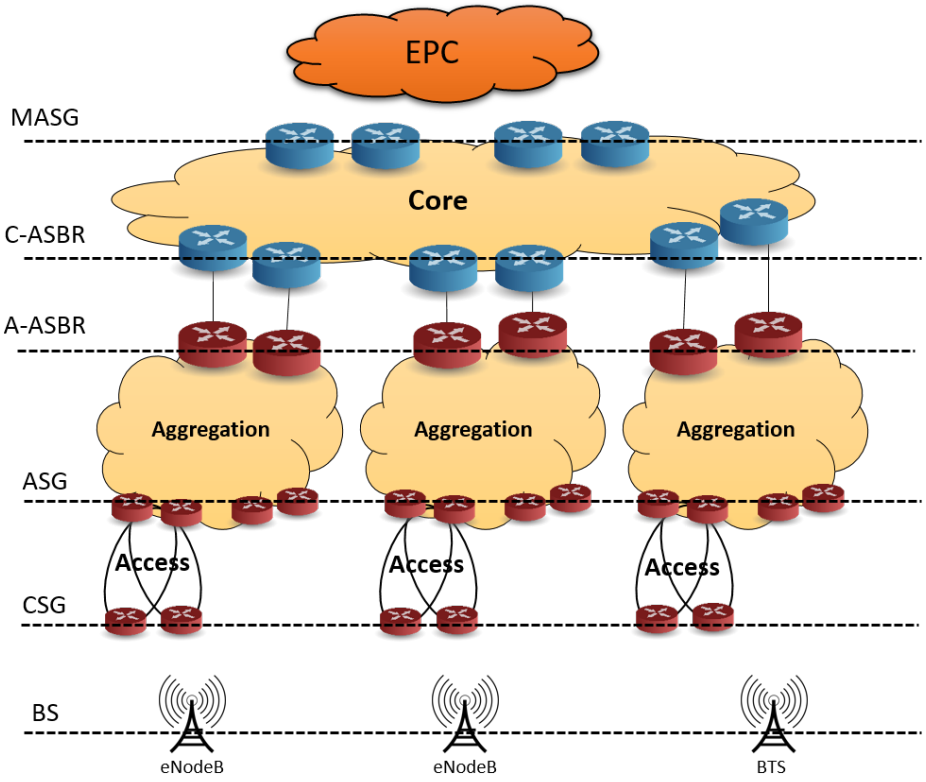


Figure 3: Structure of an IP-RAN network.

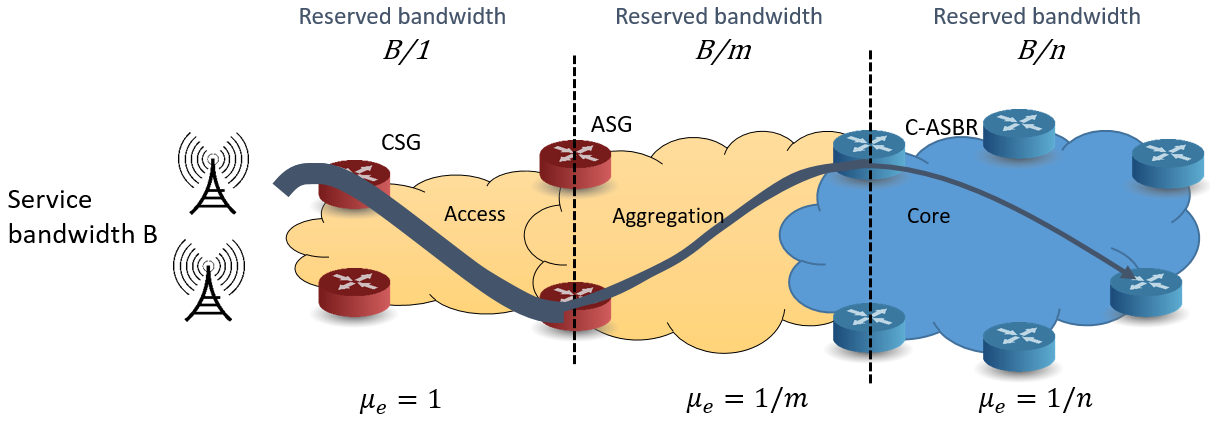


Figure 4: Convergence Ratios (CRs) in an IP-RAN network for a service with bandwidth $B$. The applied CR depends on the area of the network traversed by the service.

simplicity, we will assume that the CR factor is the same for all the nodes of a given layer, even if with some observation of the traffic, it may be possible to fine-tune the factor for different areas in the network. Note that users may explicitly state that the CR factor should not be applied to a subset of demands.

As shown in Figure 4, a service of size $B$ will consume $B$, $B/m$, and $B/n$ units of bandwidth respectively on access, aggregation and core as the convergence ratios of these levels are $\mu_e = 1$, $\mu_e = \frac{1}{m}$, and $\mu_e = \frac{1}{n}$.

In practice, as the convergence ratio can be applied to a subset of demands with the same characteristics,

it is important to respect two constraints: (1) there is enough reserved capacity to route all the services together after we apply the CR factor of the corresponding layer and (2) there is enough reserved capacity for each service alone when the CR factor is not applied. For instance, if three demands $k_1 = 2$ Gbps, $k_2 = 2$ Gbps, and $k_3 = 4$ Gbps need to be allocated in the network and $CR = \frac{1}{4}$, according to rule 1, $\frac{2+2+4}{4} = 2$ Gbps are reserved. However, when the flows are considered independently, the flow $k_3$ would not pass. Thus, the correct amount of bandwidth to be reserved is the maximum between each flow and the sum of flows scaled by the CR factor, i.e., $\max\left(\frac{2+2+4}{4}, 2, 2, 4\right) = 4$ Gbps.

Given an arc $e$ and a set of demands $K$ to be allocated, where $K_{\mathtt{C}}$ and $K_{\mathtt{NC}}$ are the two complementary subsets of demands with and without convergence ratio, the bandwidth reservation needed is given by

$$u_e(K) = \sum_{k \in K_{\mathtt{NC}}} D_k + \max\left(\mu_e \sum_{k \in K_{\mathtt{C}}} D_k, \max_{k \in K_{\mathtt{C}}} D_k\right) \tag{1}$$

where $\mu_e$ is the CR factor for the arc $e$ and $D_k$ the bandwidth requirement of service $k$.

In the literature, the convergence ratio can be also referred to as the over-provisioning ratio or overbooking ratio. For MPLS networks, it has been defined with regard to the number of tunnels being at peak rate. For instance, in [6, 11], they considered that only one tunnel at a time can be at its peak rate and that we need to accommodate all tunnels at their average rate at any time. In practice, as mentioned in [13], we may generalize this definition to the case where $k$ tunnels can be simultaneously at their peak rate. In our work, we only consider tunnel rates which are scaled by a factor determining the probability of tunnels (e.g., mobile users in IP-RAN) to be active. As we ensure that any active tunnel must be routed, we can consider the rate $D_k$ as a peak rate. We point out that, even in a scenario multi-slice, the convergence ratio applies only within each slice independently. In hard isolation, in fact, traffic fluctuations of one slice do not impact the traffic of the other slices.

### 2.5 The RSA problem

We represent a network as a directed graph $G = (V, E)$. An arc $e \in E$ represents a link of the physical network and has a cost per unit of capacity $C_e$, a transmission delay of $\lambda_e$ and a set $S^e$ of possible *slots allocation configuration*. An arc $e = (v, w) \in E$ shares the same bandwidth with its reversed arc $(w, v)$. Each slot configuration $s \in S^e$ of an arc $e$ has a total bandwidth capacity of $\xi_{es}$. For instance, an arc with 10 Gbps of available bandwidth would have six available configurations according to FlexE policy: 1 Gbps, 2 Gbps, 3 Gbps, 4 Gbps, 5 Gbps, and 10 Gbps. The set of services to provision for in the network is denoted $K$; a service must be routed from a source $s_k$ to a destination $t_k$, it requires $D_k$ amount of bandwidth, and it must experience a maximum end-to-end delay of $\Lambda_k$. For some services we can apply statistical multiplexing, while for others we cannot, and we refer to them $K_{\mathtt{C}}$ and $K_{\mathtt{NC}}$, respectively. Table 1 provides a summary of the notations used in the paper.

The *Routing and Slot Allocation* (RSA) problem consists of provisioning all services of a single slice while respecting QoS constraints and minimizing the cost of the allocated slots. For each service, a routing path must be decided, and for the entire slice, bandwidth reservations slots must be assigned on all the arcs that are used. This problem is clearly NP-hard, as it is a generalization of the discrete cost multicommodity network design problem [10].

## 3 Models and algorithms

This section presents models and algorithms for the RSA problem with statistical multiplexing. First, we propose an arc-based compact integer linear program that can be solved to optimality using CPLEX [25] on small scale instances. Then, we derive a path-based extended formulation that contains an exponential number of variables, and we describe the algorithms used to efficiently solve it using column generation. In particular, we propose a greedy algorithm and a local search algorithm to complement the column generation procedure, giving the global *FlexE-CG* algorithm.

| Symbol | Definition |
|---|---|
| $G = (V, E)$ | The network |
| $V$ | Set of routers (indexed by $v$) |
| $E$ | Set of arcs (indexed by $e$) |
| $\omega^{-}(v)$ | Set of incoming arcs of node $v$ |
| $\omega^{+}(v)$ | Set of outgoing arcs of node $v$ |
| $C_e$ | Cost of the arc $e$ |
| $\mu_e$ | Convergence ratio of the arc $e$ |
| $b_e$ | Capacity of the arc $e$ |
| $\lambda_e$ | Delay of the arc $e$ |
| $u_e(K)$ | Bandwidth required to route a set of services $K$ on the arc $e$ |
| $S^e$ | Set of slot configurations for a given arc $e$ (indexed by $s$) |
| $\xi_{es}$ | Bandwidth size of a slot configuration $s$ on arc $e$ (indexed by $s$) |
| $A(x)$ | Function giving the minimal slot configuration needed to forward $x$ amount of bandwidth |
| $K$ | Set of services in the slice (indexed by $k$) |
| $K_{\mathtt{C}}$ | Set of services impacted by statistical multiplexing (with **C**onvergence ratio) |
| $K_{\mathtt{NC}}$ | Set of services not impacted by statistical multiplexing (**N**ot with **C**onvergence ratio) |
| $D_k$ | Bandwidth size of service $k$ |
| $\Lambda_k$ | Maximum delay of the service $k$ |
| $s_k$ | Source of the service $k$ |
| $t_k$ | Destination of the service $k$ |
| $P$ | Set of paths in the network |
| $P_k$ | Set of paths between $s_k$ and $t_k$ for service $k$ respecting the maximum delay $\Lambda_k$ |

Table 1: Summary of the notations used in the paper.

## 3.1 Mathematical formulations

Both the compact and the extended formulations that we present are variants of a multi-commodity flow problem. Their main specificity lies in the needed linearization of the slot reservation constraint (1) due to statistical multiplexing and in the ordering of slots.

Recall that the bandwidth usage of an arc $e$ is equal to the sum of the non-multiplexed demands' bandwidth and the maximum between

- the sum of the multiplexed demands' bandwidth muliplied by the convergence ratio of the arc

$$\sum_{k \in K_{\texttt{NC}}} D_k + \mu_e \sum_{k \in K_{\texttt{C}}} D_k, \tag{2}$$

- and the maximum peak rate of over all multiplexed demands' bandwidth

$$\max_{k \in K_{\texttt{C}}} \sum_{k' \in K_{\texttt{NC}}} D_{k'} + D_k. \tag{3}$$

We can thus linearize Equation (1) with $|K_{\texttt{C}} + 1|$ linear equations.

### 3.1.1 Compact formulation

Let $y_{es} \in \{0,1\}$ where $y_{es} = 1$ if the slot configuration $s$ is active on the arc $e$ and $f_{ke} \in \{0,1\}$ where $f_{ke} = 1$ if the service $k$ (also called *demand* later in the paper) is routed through the arc $e$. We formulate the problem as follows:

$$\min \sum_{e \in E} C_e \sum_{s \in S^e} \xi_{es} y_{es} \tag{4a}$$

$$\text{s.t} \sum_{e:(v,v') \in E} f_{ke} - \sum_{e:(v',v) \in E} f_{ke} = \begin{cases} 1 \text{ if } v = s_k, \\ -1 \text{ if } v = t_k, \\ 0 \text{ otherwise;} \end{cases} \qquad \forall v \in V, \forall k \in K, \tag{4b}$$

$$\sum_{e \in E} \lambda_e f_{ke} \leq \Lambda_k \qquad \forall k \in K, \tag{4c}$$

$$\sum_{s \in S^e} y_{es} \leq 1 \qquad \forall e \in E, \tag{4d}$$

$$\sum_{k \in K_{\texttt{NC}}} D_k f_{ke} + \mu_e \sum_{k \in K_{\texttt{C}}} D_k f_{ke} \leq \sum_{s \in S^e} \xi_{es} y_{es} \qquad \forall e \in E, \tag{4e}$$

$$\sum_{k \in K_{\texttt{NC}}} D_k f_{ke} + D_k f_{ke} \leq \sum_{s \in S^e} \xi_{es} y_{es} \qquad \forall e \in E, \forall k \in K_{\texttt{C}}, \tag{4f}$$

$$f_{ke} \in \{0,1\} \qquad \forall k \in K, \forall e \in E^k, \tag{4g}$$

$$y_{es} \in \{0,1\} \qquad \forall e \in E, \forall s \in S^e. \tag{4h}$$

The objective function is the cost of activating the slots on the network (i.e., the sum of the capacity of activated slots on all arcs) and we want to minimize it. Inequalities (4b) are the classical flow conservation constraints, inequalities (4c) limit the delay of each path and inequalities (4d) ensure that at most one slot configuration is active per arc. Since we need to consider statistical multiplexing for the arc capacity, inequalities (4e) and (4f) translate Equation (2) and Equation (3) into arc capacity constraints.

### 3.1.2 Extended formulation

In the extended formulation, we get rid of the $f_{ke}$ variables and replace them by considering the set of paths between the source and destination of the demands. The set of variables $y$ retains the same meaning as in the compact formulation, and we introduce the set of variables $x_{kp} \in \{0,1\}$ where $x_{kp} = 1$ if service $k$ is routed through the path $p$. The path-based extended formulation is as follows:

$$\min \sum_{e \in E} C_e \sum_{s \in S^e} \xi_{es} y_{es} \tag{5a}$$

s.t

$$\delta_e : \sum_{k \in K_{\mathtt{NC}}} D_k \sum_{p \in P^k : e \in p} x_{kp} + \mu_e \sum_{k \in K_{\mathtt{C}}} D_k \sum_{p \in P^k : e \in p} x_{kp} \leq \sum_{s \in S^e} \xi_{es} y_{es} \qquad \forall e \in E \tag{5b}$$

$$\pi_{ek} : \sum_{k \in K_{\mathtt{NC}}} D_k \sum_{p \in P^k : e \in p} x_{kp} + D_k \sum_{p \in P^k : e \in p} x_{kp} \leq \sum_{s \in S^e} \xi_{es} y_{es} \qquad \forall e \in E, \forall k \in K_{\mathtt{C}} \tag{5c}$$

$$\gamma_k : \sum_{p \in P^k} x_{kp} \geq 1 \qquad \forall k \in K \tag{5d}$$

$$\theta_e : \sum_{s \in S^e} y_{es} \leq 1 \qquad \forall e \in E \tag{5e}$$

$$x_{kp} \in \{0,1\} \qquad \forall k \in K, \forall p \in P^k \tag{5f}$$

$$y_{es} \in \{0,1\} \qquad \forall e \in E, \forall s \in S^e \tag{5g}$$

Similarly to the compact formulation, inequalities (5b) and (5c) translate Equation (2) and Equation (3) into arc capacity constraints. Inequalities (5d) routes each service on at least one path, and inequalities (5e) ensure that at most one slot configuration is active per arc. The objective function is the same as the compact formulation (4). Note that $\delta_e$, $\pi_{ek}$, $\gamma_k$ and $\theta_e$ represent the dual variables associated with each constraint in the dual problem of the relaxation.

This formulation has $|V||K|$ fewer constraints than the compact formulation. As path variables $x_{kp}$ are in exponential number, we will use a column generation algorithm [16] to generate them when solving the relaxation in Section 3.2. The end-to-end delay constraint (4c) from the compact formulation, or any relevant path constraint, will be considered in the path generation, i.e., the pricing problem of the column generation procedure. After the column generation procedure finds a relaxed solution, we derive an integer solution using a randomized rounding algorithm.

## 3.2 Algorithms

Based on the extended formulation (5), we now present a column generation algorithm along with two complementary algorithms: a greedy and a local search algorithms. These algorithms are then combined to define an efficient heuristic, *FlexE-CG*.

### 3.2.1 Column generation algorithm

Column generation starts from a master problem with a subset of columns, called the *Reduced Master Problem* (RMP). To generate new columns, we obtain the dual values associated with the RMP constraints and feed them to a pricing problem that checks if any improving column exists for the current RMP. The algorithm goes back and forth between the resolution of the RMP and the (possibly multiple) pricing problem, adding

improving columns in the process. Once the pricing problem can no longer find improving columns, the procedure stops and the relaxation found is optimal.

**Pricing problem** To define the pricing problem, we use the dual of the extended formulation's relaxation (where we change the domain of variables $x_{kp}$ from $\{0,1\}$ to $\mathbb{R}$ and the domain of variables $y_{es}$ from $\{0,1\}$ to $[0,1]$). With dual variables $\delta_e$, $\pi_{ek}$, $\gamma_k$ and $\theta_e$ corresponding to their respective constraints, we formulate the dual problem as follows:

$$\max \sum_{k \in K} \gamma_k - \sum_{e \in E} \theta_e \tag{6a}$$

s.t.

$$y_{es} : \theta_e \geq \xi_{es} \left( \delta_e + \sum_{k \in K_{\mathtt{C}}} \pi_{ek} - Ce \right) \qquad \forall e \in E, \forall s \in S^e \tag{6b}$$

$$x_{kp} : \gamma_k \leq D_k \sum_{e \in p} \left( \delta_e - \sum_{k' \in K_{\mathtt{C}}} \pi_{ek'} \right) \qquad \forall k \in K_{\mathtt{NC}}, \forall p \in P^k \tag{6c}$$

$$x_{kp} : \gamma_k \leq D_k \sum_{e \in p} \left( \mu_e \delta_e - \pi_{ek} \right) \qquad \forall k \in K_{\mathtt{C}}, \forall p \in P^k \tag{6d}$$

$$\gamma \geq 0, \delta \geq 0, \theta \geq 0, \pi \geq 0 \tag{6e}$$

Finding an improving column in the *Reduced Master Problem* is equivalent to finding any violated inequality in the corresponding dual problem. In our case, these inequalities correspond to inequalities (6c) and (6d), and we need to find, for any service $k \in K_{\mathtt{NC}}$ if there exists any path $p$ such that

$$D_k \sum_{e \in p} \left( \delta_e - \sum_{k' \in K_{\mathtt{C}}} \pi_{ek'} \right) < \gamma_k$$

or, for any service $k \in K_{\mathtt{C}}$, if there exists any path $p$ such that

$$D_k \sum_{e \in p} \left( \mu_e \delta_e - \pi_{ek} \right) < \gamma_k$$

We can see, in both cases, that proving the (non-)existence of an improving column for a demand corresponds to computing its shortest path. Since demands have delay constraints, the pricing problem thus corresponds to the resource-constrained shortest path problem. The weight of an arc $e$ depends on the type of demand: for a CR demand, it is given by $D_k \left( \mu_e \delta_e - \pi_{ek} \right)$; for a NC demand, it is given by $D_k \left( \delta_e - \sum_{k' \in K_{\mathtt{C}}} \pi_{ek'} \right)$. Since we need to satisfy an end-to-end delay constraint, we use the LARAC [26] algorithm. When multiple additive constraints need to be considered (e.g., delay, jitter, loss), GEN-LARAC [32] can also be used.

**Randomized rounding** Once we obtain an optimal solution for the extended formulation's relaxation and a subset of path $\bar{P}_k$ for each demand, we derive an integer solution using a randomized rounding algorithm. The algorithm selects, for each demand $k$, a path $p$ from $\bar{P}_k$ at random with the probability

$$P\left(k \text{ is routed on } p\right) = \frac{x^*_{kp}}{\sum_{p' \in \bar{P}_k} x^*_{kp'}}$$

where $x^*_{kp}$ is the value of $x_{kp}$ in the optimal solution of the relaxation of Formulation 5. If selecting a path violates some arc capacity constraints, we remove it from the set of available paths and redraw a path at random until we either find a valid path or there are no remaining paths. The final step consists of greedily routing the remaining demands without a selected path using the greedy algorithm presented in Algorithm 1. The algorithm can be run several times in parallel, and we keep the feasible solution with minimum value.

### 3.2.2 Greedy algorithm

We propose a greedy procedure, described in Algorithm 1, that we use for warm starting the column generation algorithm, i.e, to give an initial set of columns, and for finding missing paths in the randomized rounding algorithm. It successively finds the shortest path constrained by an end-to-end delay; At each iteration, the weights of the arcs depend on the slot allocation already performed for the previous paths selected. More precisely, the weight of an arc $e$ when we calculate the path for a demand $k$ is given by

$$w^k(e) = \begin{cases} 1 \text{ if } A\left(u_e\left(K_e\right)\right) \geq u_e\left(K_e \cup \{k\}\right), \\ 1 + C_e \text{ otherwise.} \end{cases}$$

where $u_e\left(K\right)$ gives the bandwidth used by the set of demand $K$ on the arc $e$ and $A\left(x\right)$ gives the slot allocation needed for $x$ Gbps amount of bandwidth, i.e.,

$$A(x) = \begin{cases} \lceil x/1 \rceil \text{ if } x < 5, \\ \lceil x/5 \rceil * 5 \text{ otherwise.} \end{cases}$$

This weight function favors routing on arcs where allocated resources can be used *for free*.

**Algorithm 1** Greedy algorithm

**Input:** A network $G = (V, E)$, arc capacity $b_e$, $\forall e \in E$, set of demands $K$ to route, set of demands $K_e$ on each arc $e$
**Output:** Set of paths $P$

1: $P \leftarrow \emptyset$
2: $K_e \leftarrow \emptyset$, $\forall e \in E$
3: **for** demand $k \in K$ **do**
4: $\quad E^k \leftarrow \{e : u_e\left(K_e \cup \{k\}\right) \leq b_e\}$
5: $\quad w^k(e) = \begin{cases} 1 \text{ if } A\left(u_e\left(K_e\right)\right) \geq u_e\left(K_e \cup \{k\}\right) \\ 1 + C_e \text{ otherwise} \end{cases} \quad \forall e \in E^k$
6: $\quad$ Build weighted graph $G^k = (V, E^k, w^k)$
7: $\quad$ Find shortest path $p$ from $s_k$ to $t_k$ in $G^k$
8: $\quad P \leftarrow P \cup \{p\}$
9: $\quad$ **for** arc $e \in p$ **do**
10: $\quad\quad K_e \leftarrow K_e \cup \{k\}$
**return** $P$

### 3.2.3 Local search algorithm

Finally, we design a local search algorithm, described in Algorithm 2, to improve existing solutions. At each step, the algorithm selects an arc and decreases its bandwidth to try to improve the current solution. It removes the demands that were previously using this arc and tries to reroute them. It reverts to the previous solution if it can't reroute them or if the new solution has a higher cost. When the algorithm has evaluated all the arcs, it stops and returns the best solution found.

We chose to rank the arcs based on the amount of unused bandwidth multiplied by the arc's cost (see line (4)) because it is rare for all allocated slots to be fully filled in a given solution. Thus, we can hope to redirect traffic and reduce some slot allocation to the previous ones. For instance, an arc transmitting 5.1 Gpbs requires a 10 Gbps slot configuration, wasting 4.9 Gbps. By limiting the capacity of the arc to 5 Gbps, we could find new paths for the demands using this arc without requiring a bigger slot allocation somewhere else: the new solution would thus be cheaper.

**Algorithm 2** Local search algorithm

**Input:** A network $G = (V, E)$, arc capacity $b_e \forall e \in E$, set of demands K, set of paths $P = \{p_k : \forall k \in K\}$
**Output:** Set of paths $P$

$E_{\texttt{CAND}} \leftarrow E$
**while** $E_{\texttt{CAND}} \neq \emptyset$ **do**
  $K_e \leftarrow \{k : e \in p_k\} \quad \forall e \in E$
  $\tilde{e} \leftarrow \arg\max_{e \in E_{\texttt{CAND}}} C_e \times (S(u(e, K_e)) - u(e, K_e))$
  $E_{\texttt{CAND}} \leftarrow E_{\texttt{CAND}} \setminus \tilde{e}$
  $(P_{\texttt{OLD}}, K_{\texttt{OLD}}, b_{\texttt{OLD}}) \leftarrow (P, K_{\tilde{e}}, b_{\tilde{e}})$
  $b_{\tilde{e}} \leftarrow \lfloor S(u(\tilde{e}, K_{\tilde{e}})) \rfloor$
  **for** demand $k \in K_{\texttt{OLD}}$ **do**
    **for** arc $e \in p_k$ **do**
      $K_e \leftarrow K_e \setminus k$
    $p_k \leftarrow \emptyset$
  $P_{\texttt{NEW}} \leftarrow P \cup Greedy(G, K_{\texttt{OLD}}, K_e, b_)$
  **if** $Cost(P_{\texttt{NEW}}) < Cost(P_{\texttt{OLD}})$ **then**
    $P \leftarrow P_{\texttt{NEW}}$
  **else**
    $P \leftarrow P_{\texttt{OLD}}$
    Restore $(P_{\texttt{OLD}}, K_{\texttt{OLD}}, b_{\texttt{OLD}})$ as current solution
**return** P

### 3.3 Overall architecture of the *FlexE-CG* algorithm

Figure 5 depicts the overall architecture of *FlexE-CG*, the heuristic algorithm we propose. As illustrated, the column generation routine can be warm-started by providing a first initial feasible solution found by the greedy algorithm. The pricing problems corresponding to all demands can be solved in parallel. And in the rounding step, several randomized rounding routines can be executed in parallel. Finally, the best solution among those provided in the rounding step is selected. And the local search algorithm is applied to improve the solution.

This algorithm can be implemented inside a powerful SDN controller to calculate FlexE slot reservations over physical arcs to satisfy the QoS requirements of demands in a given slice.

## 4 Improving the lower bound

The relaxed solution given by the column generation algorithm provides a lower bound to the RSA problem. This bound is useful to measure the quality of any solution to the original problem, especially on larger instances where the compact formulation cannot be solved in a reasonable amount of time.

In this section, we reinforce the extended formulation with valid inequalities to improve the lower bound. First, we propose an extension of the well-known cover inequalities [5] associated with the capacity constraints. Second, thanks to the max-flow min-cut theorem, we propose families of inequalities based on the edge-cut in the graph. Third, we propose the 1-slot inequalities that focus on slot configurations activation for a given edge-cut. Finally, we analyze the strength of these valid inequalities using a polyhedral analysis and we propose algorithms to generate them.

Before presenting the inequalities, let us lay down some notations summarized in Table 2. For a given arc $e$, the function $S^e_>(z)$ (resp. $S^e_\geq(z)$, $S^e_<(z)$, $S^e_\leq(z)$) provides the set of slot configurations with a bandwidth reservation greater than $z$ (resp., greater than or equal to, less than, less than or equal to $z$). We denote by $s^e_{\max}$ (resp. $s^e_{\min}$) the slot configuration with the maximum (resp. minimum) bandwidth size in $S^e$. The size

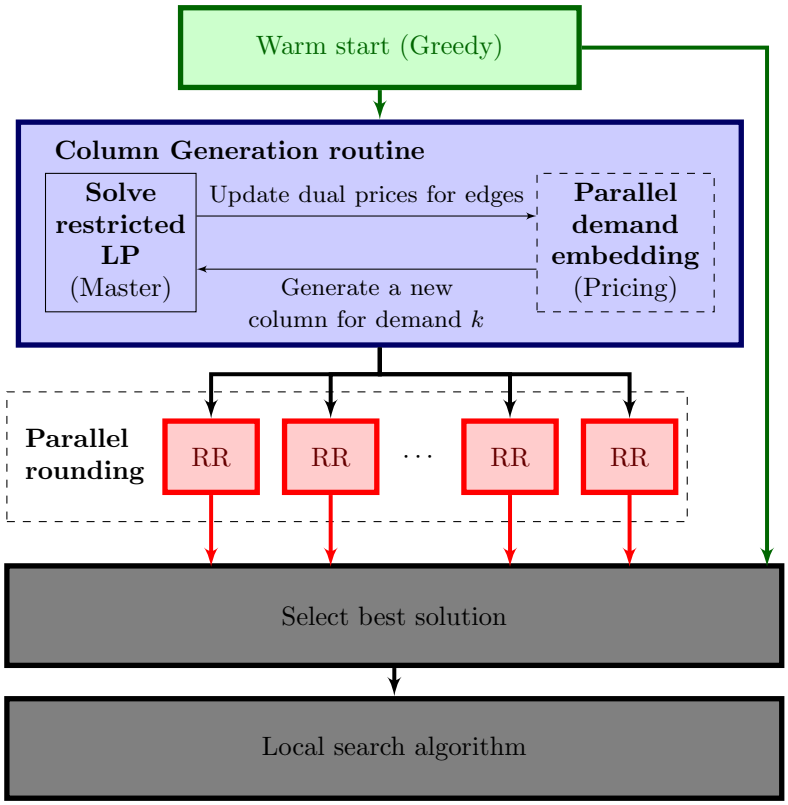


Figure 5: Architecture of *FlexE-CG*, a column-generation-based heuristic using parallelization and warm start.

| Symbol | Signification |
|---|---|
| $S^e_>(z)$ | set of slot configurations with size greater than $z$ |
| $S^e_\geq(z)$ | set of slot configurations with size greater than or equal to $z$ |
| $S^e_<(z)$ | set of slot configurations with size less than $z$ |
| $S^e_\leq(z)$ | set of slot configurations with size less than or equal to $z$ |
| $s^e_{\max}$ | slot configuration of maximum size in $S^e$ |
| $s^e_{\min}$ | slot configuration of minimum size in $S^e$ |
| $s^e_S$ | slot configuration of maximum size in $S$ |
| $\xi_{e\max}$ | size of the slot configuration associated with $s^e_{\max}$ |
| $\xi_{e\min}$ | size of the slot configuration associated with $s^e_{\min}$ |
| $\xi_{eS}$ | size of the slot configuration of maximum size in $S$ |
| $\delta(V')$ | arcs between $V'$ and $V \setminus V'$ |
| $\mu^{\delta(V')}_{\min}$ | smallest value of $\mu^e$ where $e \in \delta(V')$ |
| $K(V')$ | set of demands such that both end-points are on the opposite side of the cut |
| $K_{\mathtt{C}}(V')$ | set of demands with convergence ratio |
| $K_{\mathtt{NC}}(V')$ | set of demands without convergence ratio |

Table 2: Summary of the notations used when improving the lower bound.

of the slot configuration associated with $s^e_{\max}$ (resp. $s^e_{\min}$) is denoted $\xi_{e\max}$ (resp. $\xi_{e\min}$) and the associated variables $y_{e\max}$ (resp. $y_{e\min}$).

By abuse of language, $\xi_{eS}$ is the size of the slot configuration of maximum size in a given set of slots $S$ (i.e., $\xi_{eS} = \arg\max_{s\in S} \xi_{es}$), and $s^e_S$ is the slot configuration of maximum size in a given set of slots $S$. Remark that $\xi_{es^e_{\max}} = \xi_{e\max}$.

Let $V' \subset V$ be an edge-cut of the network, and we denote by $\delta(V')$ the arcs between $V'$ and $V \setminus V'$ (i.e., $E[V', V \setminus V']$). We denote by $\mu^{\delta(V')}_{\min}$ the smallest convergence ratio $\mu_e$ of all the arcs in the edge-cut $\delta(V')$, where $e \in \delta(V')$. Let $K(V') \subseteq K$ be the set of demands such that both end-points are on the opposite side of the cut. $K_{\mathtt{C}}(V') \subseteq K(V')$ and $K_{\mathtt{NC}}(V') \subseteq K(V')$ denote the set of demands with and without convergence ratio, respectively.

## 4.1 Valid inequalities

In this section, we introduce three families of inequalities. First, we propose an extension of the well-known cover inequalities associated with the capacity constraints. Second, thanks to the flow and edge-cut relationship, we propose inequalities for all possible edge-cuts in the graph. Third, we propose 1-slot inequalities for all edge-cuts based on slot configurations.

### 4.1.1 Extension of cover inequalities

Cover inequalities consider a subset of demands $K' \subseteq K$ such that all these demands cannot fit on a given arc $e \in E$. We extend these cover inequalities to consider the different slot configurations on FlexE arcs.

For a subset of demands $K' \subseteq K$ and an arc $e \in E$, the slot configuration $s \in S^e$ is invalid for $K'$ if and only if

$$\sum_{k\in K_{\mathtt{NC}}\cap K'} D_k \sum_{p\in P^k: e\in p} x_{kp} + \mu_e \sum_{k\in K_{\mathtt{C}}\cap K'} D_k \sum_{p\in P^k: e\in p} x_{kp} > \xi_{es} y_{es} \tag{7a}$$

or

$$\sum_{k'\in K_{\mathtt{NC}}\cap K'} D_{k'} \sum_{p\in P^{k'}: e\in p} x_{k'p} + D_k \sum_{p\in P^k: e\in p} x_{kp} > \xi_{es} y_{es}, \qquad \forall k \in K_{\mathtt{C}} \cap K'. \tag{7b}$$

For each slot configuration $s \in S^e$ that is invalid for a given $K'$, the following inequality is valid

$$\sum_{k' \in K'} \sum_{p \in P^{k'}: e \in p} x_{k'p} \leq |K'| - 1 + \sum_{s' \in S^e_>(\xi_{es})} y_{es'}.$$

#### 4.1.2 Edge-cut inequalities

For a given edge-cut, we derive two families of inequalities based on the relationship between the edge-cut and the flow. Indeed, for a given edge-cut in a graph, the minimum amount of traffic that crosses the cut is equal to the sum of bandwidth for all demands originating on one side and terminating on the other side. Due to the convergence ratio, we can derive two kinds of edge-cut inequalities. The first family considers the convergence ratio in the flow crossing the edge-cut, whereas the second family considers the convergence ratio on the size of the allocated slots. We show an example that each of these two families of inequalities can dominate the other one, depending on the convergence ratio of each arc. Thus, the two families of inequalities are useful to strengthen the linear relaxation.

***Ratio flow* inequalities** Let $D^r(V')$ be the minimum amount of traffic crossing the edge-cut $V'$ considering the convergence ratio, i.e.,

$$D^r(V') = \sum_{k \in K_{\text{NC}}(V')} D_k + \max \left( \mu_{\min}^{\delta(V')} \sum_{k \in K_{\text{C}}(V')} D_k, \max_{k' \in K_{\text{C}}(V')} (D_{k'}) \right).$$

Due to the relationship between the edge-cut and the flow, we know that the traffic $D^r(V')$ must cross the arcs of the edge-cut $V'$. We can deduce the following inequality

$$\sum_{e \in \delta(V')} \sum_{s \in S^e} \xi_{es} y_{es} \geq \lceil D^r(V') \rceil. \tag{8}$$

Indeed, due to the slots' configuration, if $D^r(V')$ is fractional then it must be rounded up to the smallest slot reservation (in our case, 1Gbps).

***Ratio capacity* inequalities** Let now $D(V')$ be the minimum amount of bandwidth for all demands that cross the edge-cut $V'$ at least once. Formally, $D(V') = \sum_{k \in K(V')} D_k$ where $K(V') \subseteq K$ is the set of demands such that their source and destination are not together in $V'$ or in $V \setminus V'$.

Due to the relationship between the edge-cut and the flow, we know that the traffic $D(V')$ must cross the arcs of the edge-cut $V'$ up to the smallest slot size possible (in our case, 1Gbps). We can deduce the following inequality

$$\sum_{e \in \delta(V')} \sum_{s \in S^e} \frac{1}{\mu^e} \xi_{es} y_{es} \geq \lceil D(V') \rceil. \tag{9}$$

Indeed, due to the slot configurations if $D(V')$ is fractional then $D(V')$ must be rounded up.

**Example** Let us consider an edge-cut $V'$ composed of two arcs $\{e_1, e_2\}$ with the same possible slot configurations of 1Gpbs, 2Gpbs, and 3Gpbs; the arc $e_1$ has a convergence ratio of $\mu_{e_1} = 1$ and the arc $e_2$ has a convergence ratio $\mu_{e_2} = \frac{1}{2}$. We consider 3 demands crossing the cut: $k_1$, $k_2$ and $k_3$, each requiring 1.1 Gbps.

In the case where all demands require statistical multiplexing, the amount of traffic crossing the cut $D(V')$ is equal to 3.3 Gbps while the amount of scaled traffic $D^r(V')$ is equal to 1.65 Gpbs ($\frac{1}{2} \times 3.3$ Gbps). The ratio flow edge-cut inequality is

$$1y_{e_1 1} + 2y_{e_1 2} + 3y_{e_1 3} + 1y_{e_2 1} + 2y_{e_2 2} + 3y_{e_2 3} \geq \lceil 1.65 \rceil = 2, \tag{10}$$

the ratio capacity edge-cut inequality is

$$1y_{e_1 1} + 2y_{e_1 2} + 3y_{e_1 3} + 2y_{e_2 1} + 4y_{e_2 2} + 6y_{e_2 3} \geq \lceil 3.3 \rceil = 4. \tag{11}$$

The latter dominates the former. Indeed, inequality (10) is equivalent to

$$2y_{e_1 1} + 4y_{e_1 2} + 6y_{e_1 3} + 2y_{e_2 1} + 4y_{e_2 2} + 6y_{e_2 3} \geq 4. \tag{12}$$

In the case where no demand requires statistical multplexing, the amount of original traffic and scaled traffic are both equal to 3.3Gbps. The ratio flow inequality is

$$1y_{e_1 1} + 2y_{e_1 2} + 3y_{e_1 3} + 1y_{e_2 1} + 2y_{e_2 2} + 3y_{e_2 3} \geq \lceil 3.3 \rceil = 4, \tag{13}$$

which dominates the following ratio capacity edge-cut inequality

$$1y_{e_1 1} + 2y_{e_1 2} + 3y_{e_1 3} + 2y_{e_2 1} + 4y_{e_2 2} + 6y_{e_2 3} \geq \lceil 3.3 \rceil = 4. \tag{14}$$

Thus, the two families of inequalities can improve the linear relaxation.

#### 4.1.3 1-slot inequalities

Our last type of inequalities is derived from edge-cut inequalities and focuses on slot configuration activation. For a given edge-cut, there exist subsets of slot configurations for its arcs, called *1-slot sets*, such that when none of them are active, the capacity of the edge-cut does not satisfy the flow through the edge-cut.

Following our previous example, if all three demands do not require statistical multiplexing, 3.3 Gbps cross the edge-cut. Since the two arcs provide 1 Gbps, 2 Gbps or 3 Gbps configurations, one of the possible *1-slot set* contains the 2 and 3 Gbps configurations for both arcs. Indeed, with only 1 Gbps on each arc, the edge-cut capacity is insufficient.

More formally, let $\mathcal{S}^{\mathbb{1}}_{V'}$ be the set of *1-slot set* of the edge-cut $V'$. Any *1-slot set* $S^{\mathbb{1}} \in \mathcal{S}^{\mathbb{1}}_{V'}$ must satisfy either

$$\sum_{e \in \delta(V')} \max_{s \in S^e \setminus S^{\mathbb{1}}} \xi_{es} < \lceil D^r(V') \rceil, \tag{15}$$

or

$$\sum_{e \in \delta(V')} \frac{1}{\mu^e} \max_{s \in S^e \setminus S^{\mathbb{1}}} \xi_{es} < \lceil D(V') \rceil. \tag{16}$$

The 1-slot inequality defined by

$$\sum_{e \in \delta(V')} \sum_{s \in S^e \cap S^{\mathbb{1}}} y_{es} \geq 1 \qquad \forall V' \subset V, \forall S^{\mathbb{1}} \in \mathcal{S}^{\mathbb{1}}_{V'} \tag{17}$$

is a valid inequality of Problem (5).

**Example** From the previous example, where the arc $e_1$ has a convergence ratio of 1 and the arc $e_2$ has a convergence ratio $\mu^e = \frac{1}{2}$. By considering $K_{\mathtt{C}} = \{d_1, d_2, d_3\}$ we deduce the following 1-slot inequality

$$y_{e_1 1} + y_{e_1 2} + y_{e_1 3} + y_{e_2 2} + y_{e_2 3} \geq 1 \tag{18}$$

and when $K_{\mathtt{NC}} = \{d_1, d_2, d_3\}$ we deduce the following 1-slot inequality

$$y_{e_1 1} + y_{e_1 2} + y_{e_1 3} + y_{e_2 3} \geq 1 \tag{19}$$

We can remark that this family of inequalities has a lot of symmetries. Indeed, the inequality (18) is also valid by shifting the arcs on the edge-cut.

## 4.2 Relationship with state of the art

While classical cover inequalities are well studied [5], the constraints of the RSA problem, notably the multiplexing, leads to difference with the proposed inequalities found in the literature. Frangioni et al. [21] propose an adaptation of the cover inequalities for the capacitated network design problem that considers the first slot configuration. Our extension of cover inequalities handles slot configurations (not only the first one) and multiplexing.

Our edge-cut inequalities can be seen as generalizations and specializations in the case of multiplexing of the inequalities found in [30, 22]. Rack et al. [30] propose a particular case of our edge-cut inequalities where slots have the same size and no multiplexing is considered. The authors also propose other valid inequalities, mixing ideas from cover and edge-cut inequalities. But the major drawback of these inequalities, called *residual capacity* cuts, is the destruction of the pricing problem as they involve several arcs in the same inequality. For a model based on path variables, this kind of inequalities prevents the resolution of the pricing problem with a pseudo polynomial time path computation algorithm. Our edge-cut and 1-slot inequalities do not contain path variables, and thus the pricing problem remains the same. And even though our cover inequalities contain path variables, they do not impact the structure of the pricing problem as they are defined for each arc, which leads to only one additional dual cost. Gabrel et al. [22] propose a generic family of inequalities using a subset of arcs. A particular case of this family of inequalities is the edge-cut inequalities without considering multiplexing.

## 4.3 Separation problems and algorithms

We now describe the algorithms we used to separate the inequalities.

### 4.3.1 Cover inequalities

As a reminder, for an arc $e$ and a subset $K'$ of demands satisfying Equations (7), a valid cover inequalities of *FlexE-CG* has the following form

$$\sum_{k' \in K'} \sum_{p \in P^{k'}: e \in p} x_{k'p} \leq |K'| - 1 + \sum_{s' \in S^e_>(s)} y_{es'}.$$

The goal of this separation problem is to find, from a fractional solution $(x^*, y^*)$ of *FlexE-CG*, a subset of demands $K'$ violating any multiplexed capacity constraints for a given slot configuration $s$. It is equivalent to finding any subset $K'$ such that

$$\sum_{k' \in K'} \sum_{p \in P^{k'}: e \in p} (x^*_{k'p} - 1) > \sum_{s' \in S^e_>(s)} y^*_{es'} - 1.$$

The problem is NP-Hard. We thus propose an ILP formulation to separate the inequalities exactly, as well as an adaptation of a classic cover separation heuristic [31] to improve our convergence time.

**ILP** We first formulate the separation problem for cover inequalities for a given arc $e$ and a given slot configuration $s$ as an ILP. We have three types of variables: the set of variables $z_k$, for $k \in K$ that indicates which demands are selected; the variable $a$ that indicates if the classical capacity constraint is violated; and the set of variables $b_k$, for $k \in K_{\mathtt{C}}$, and that indicates which multiplexed capacity constraints are violated. The formulation is as follows:

$$\min \sum_{k \in K} \sum_{p \in P^k : e \in p} (1 - x^*_{kp}) z_k \tag{20a}$$

$$s.t. \sum_{k \in K_{\mathtt{NC}}} D_k z_k + \mu^e \sum_{k \in K'_{\mathtt{C}}} D_k z_k > \xi_{es} a \tag{20b}$$

$$\sum_{k' \in K_{\mathtt{NC}}} D_{k'} z'_k + D_k z_k > \xi_{es} b_k \qquad \forall k \in K_{\mathtt{C}} \tag{20c}$$

$$a + \sum_{k \in K_{\mathtt{C}}} b_k \geq 1 \tag{20d}$$

$$z_k \in \{0, 1\} \qquad \forall k \in K \tag{20e}$$

$$a \in \{0, 1\} \tag{20f}$$

$$b_k \in \{0, 1\} \qquad \forall k \in K \tag{20g}$$

The objective minimizes the sum of the set of selected demands and their corresponding relaxed values $x^*$. If this value is strictly less than

$$1 - \sum_{s' \in S^e_>(s)} \xi_{es'} y^*_{es'},$$

we found a violated cover inequality that we can add to the problem. Constraints (20b) and constraints (20c) allow us to determine which capacity constraints are violated. Constraints (20d) ensure that at least we violated one capacity constraints (classical or multiplexed).

**Heuristic** Our separation heuristic, presented in Algorithm 3, for the cover inequalities is based on a heuristic proposed in [31]. In the same fashion, we sort the demands in increasing order by the ratio between their cost $(1 - x^*_k)$ and their size $D_k$ and greedily select them by decreasing order until we reach the capacity reservation of the slot configuration.

However, in our case, due to statistical multiplexing, the size of a demand depends on other selected statistically multiplexed demands $k \in K_{\mathtt{C}}$. We thus need to maintain information about which statistical-multiplexed demands were already selected (see lines 4-7). Then, we recompute the size of each statistically multiplexed every time we need to select the next best demand according to the cost-to-size ratio (see line 10).

#### 4.3.2 1-slot inequalities

As a reminder, 1-slot inequalities are defined as follows (see Section 4.1.3):

$$\sum_{e \in \delta(V')} \sum_{s \in S^e_{\geq}(S_{V'}(e))} y_{es} \geq 1, \forall V' \subseteq V$$

The separation problem consists of finding an edge-cut and a subset of slot configurations (i.e., $S_{V'}(e)$) for each arc of the cut such that downgrading at least one arc to the previous configuration (i.e., the biggest configuration smaller than the current one) would tip the edge-cut capacity below the sum of the demand going through it. Unfortunately, even when the edge-cut is given, the problem is NP-Hard (one can easily reduce the knapsack problem to our problem). We thus provide an ILP formulation to solve the separation problem and a heuristic based on the LARAC algorithm [26] for the constrained shortest path problem.

Note that we can further divide the separation problem into two regarding whether Equation (15) or Equation (16) are satisfied, i.e., whether we are in the ratio flow or the ratio capacity variant of the separation problem.

**Algorithm 3** Greedy separation algorithm for cover inequalities

**Input:** A set of demands $K = K_{\texttt{NC}} \cup K_{\texttt{C}}$, a slot configuration $(e, s)$ and its convergence ratio $\mu_e$, a fractional solution $(x^*, y^*)$
**Output:** A subset $M \subseteq K$

$x^*_k \leftarrow \sum_{p \in P_k} x^*_{pk} \forall k \in K$
Sort $K_{\texttt{NC}}$ by increased order of $(1 - x^*_k)/D_k$ into $L$
$M \leftarrow \emptyset$
$\texttt{CR}_{\texttt{SUM}} \leftarrow 0$
$\texttt{CR}_{\texttt{MAX}} \leftarrow 0$
$\texttt{NC}_{\texttt{SUM}} \leftarrow 0$
**while** $\texttt{NC}_{\texttt{SUM}} + \max(\texttt{CR}_{\texttt{SUM}}, \texttt{CR}_{\texttt{MAX}}) \leq \xi_{es}$ **do**
    $k^{\texttt{NC}} \leftarrow L.pop()$
    $\Delta_{\texttt{CR}}(k) \leftarrow \max(\texttt{CR}_{\texttt{SUM}} + \mu_e D_k, \max(\texttt{CR}_{\texttt{MAX}}, D_k)) - \max(\texttt{CR}_{\texttt{SUM}}, \texttt{CR}_{\texttt{MAX}})$
    $k^{\texttt{CR}} \leftarrow \arg\min_{k \in K_{\texttt{C}} \setminus M} \frac{(1-x^*_k)}{\Delta_{\texttt{CR}}(k)}$
    **if** $(1 - x^*_{k^{\texttt{NC}}})/D_{k^{\texttt{NC}}} \leq (1 - x^*_{k^{\texttt{CR}}})/\Delta_{\texttt{CR}}(k^{\texttt{CR}})$ **then**
        $M \leftarrow M \cup \{k^{\texttt{NC}}\}$
        $\texttt{NC}_{\texttt{SUM}} \leftarrow \texttt{NC}_{\texttt{SUM}} + D_{k^{\texttt{NC}}}$
    **else**
        $M \leftarrow M \cup \{k^{\texttt{CR}}\}$
        $K_{\texttt{C}} \leftarrow K_{\texttt{C}} \setminus \{k^{\texttt{CR}}\}$
        $\texttt{CR}_{\texttt{SUM}} \leftarrow \texttt{CR}_{\texttt{SUM}} + \mu_e D_{k^{\texttt{CR}}}$
        $\texttt{CR}_{\texttt{MAX}} \leftarrow \max(\texttt{CR}_{\texttt{MAX}}, D_{k^{\texttt{CR}}})$
**return** $M$

**ILP** We formulate the separation problem for a given edge-cut $\delta(V')$. We define a set of variables $z_{es}, \forall e \in \delta(V'), \forall s \in S^e$ that indicates if a slot configuration and its upgrades (i.e., $s \in S^e_{\geq}(\xi_{es})$) belong to the *1-slot* set. The whole formulation is as follows:

$$\min \sum_{e \in \delta(V')} \sum_{s \in S^e} \sum_{s' \in S^e_{\geq}(\xi_{es})} y^*_{es} z_{es} \tag{21a}$$

$$\sum_{s \in S^e} z_{es} \geq 1 \qquad \forall e \in \delta(V') \tag{21b}$$

$$\sum_{e \in \delta(V')} \sum_{s \in S^e} \xi_{es-1} z_{es} < D^r(V') \tag{21c}$$

The objective function (21a) minimizes the fractional value of the selected set: if it goes below 1, we have a violated 1-slot cut to add to our master problem. Since $z_{es}$ represents the slot configuration $s$ and all its upgrades, we need to sum over all of their fractional values. Constraint (21b) sets a slot configuration for each arc of the cut. Constraint (21c) ensures that the sum of the capacity of the slot configuration not in the *1-slot* set is below the flow going through the edge-cut. In the ratio capacity variant of the separation problem, this constraint is replaced by

$$\sum_{e \in \delta(V')} \frac{1}{\mu_e} \sum_{s \in S^e} \xi_{es-1} z_{es} < D(V'). \tag{21d}$$

**Heuristic** To improve the convergence time, we designed a heuristic to solve the separation problem as a constrained shortest path problem. Once the instance of the constrained shortest path problem is built,

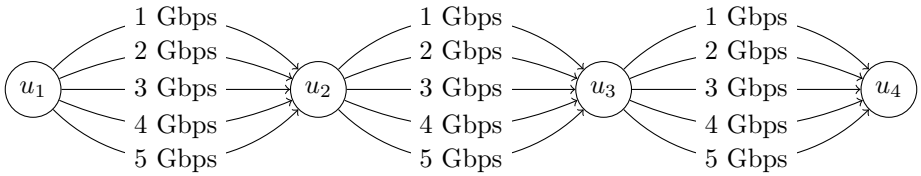


Figure 6: Example of the path-graph used for solving the one-slot separation problem on a 3-edge-cut ($e_0$, $e_1$, $e_2$) with 5 Gbps arcs and thus 5 slot configurations each. Each outgoing arc of a node $s_i$ correspond to a slot configuration available for arc $e_i$. Any path between $u_0$ and $u_4$ represents a set of configurations for the edge-cut.

we then use the LARAC algorithm [26] to get a solution. For a given edge-cut $V'$, we build a path-graph with multiple arcs between the nodes. The total number of nodes is equal to $|\delta(V')| + 1$ nodes and the total number of arcs is equal to $\sum_{e \in \delta(V')} |S^e|$. Each node $u_i$ is connected to the node $u_{i+1}$ by as many arcs as the number of slot configurations available on the i$^{\text{th}}$ arc of $\delta(V')$. Any path from $u_0$ to $u_{|\delta(V')|}$ represents a set of slot configurations for the edge-cut $V'$.

To find a 1-slot set, we need to associate with each arc, a cost, and a resource. The cost of the arc is equal to $\sum_{s' \in S^e_{\geq}(\xi_{es})} y^*_{es}$ which corresponds to the sum of the fractional values of the slot configuration associated with the arc and its upgrades (i.e., slot configurations with a higher bandwidth capacity). The resource corresponds to the bandwidth capacity of the slot configuration, i.e., $\xi_{es-1}$. We limit the resource possible to $D(V')$. Finding a violated 1-slot cut is thus equivalent to finding a constrained shortest path on this path-graph that consumes strictly less than $D^r(V')$, or $D(V')$ for ratio 1-slot, resource and costs strictly less than 1. Figure 6 shows an example of the path-graph built from a three-arcs cut.

## 4.4 Putting it all together

Putting all these separation algorithms together gives us the *allCuts* procedure to separate *FlexE-CG*. When we add the procedure on top of the *FlexE-CG* algorithm presented in Section 3.3, we called this algorithm *FlexE-CG*(*allCuts*), which is described in Figure 7. It alternates between the column generation algorithm and the separation problems procedure until no improving columns and no violated inequalities can be found. If a timeout is needed, we can stop the algorithm in between the two phases.

Before running the separation procedure, we generate a subset of edge-cut of the network that we use for all separation problems but the cover one. IP-RAN networks have a very hierarchical structure, and we exploit this to select our edge-cuts. First, we select edge-cut between the different layers of the network (e.g., between access and aggregation). Then, we select edge-cuts that separate the CSG source node (i.e., a node from where traffic originates) from the rest of the network. These nodes have mostly a degree of 1 or 2; we can easily extend these inequalities by adding the neighbors into the cut. For instance, a source node $v_1$ connected to $v_2$ and $v_3$ gives the edge-cut $(v_1, V \setminus \{v_1\})$ that can be extended to $(\{v_1, v_2\}, V \setminus \{v_1, v_2\})$, $(\{v_1, v_3\}, V \setminus \{v_1, v_3\})$ and $(\{v_1, v_2, v_3\}, V \setminus \{v_1, v_2, v_3\})$. More formally, from a cut $(V', V \setminus V')$, $\forall v \in V \setminus V'$ with a degree less than 3, we create an edge-cut $(V' \cup v)$.

The procedure searches for violated inequalities in the following order: ratio flow edge-cut inequalities, ratio capacity edge-cut inequalities, 1-slot inequalities with the ratio flow variant, 1-slot inequalities with the ratio capacity variant, and finally the cover inequalities. We do make a distinction between the two variants of the 1-slot inequalities, namely the ratio flow and ratio capacity 1-slot inequalities. For each inequality, we alternate between solving the master problem and the separation algorithm until we no longer find violated inequalities. If no violated inequalities can be found for a given cut, we go to the next separation problem in the list. Since our separation problems are NP-hard, we prioritize the use of the heuristics to solve them. If a heuristic cannot find any violated inequalities, we use the exact methods to find any. The procedure stops when we cannot find any violated inequalities with the exact methods. Once both column and cut generation are done, we run the rounding algorithm and the local search algorithm; if the row and column generation

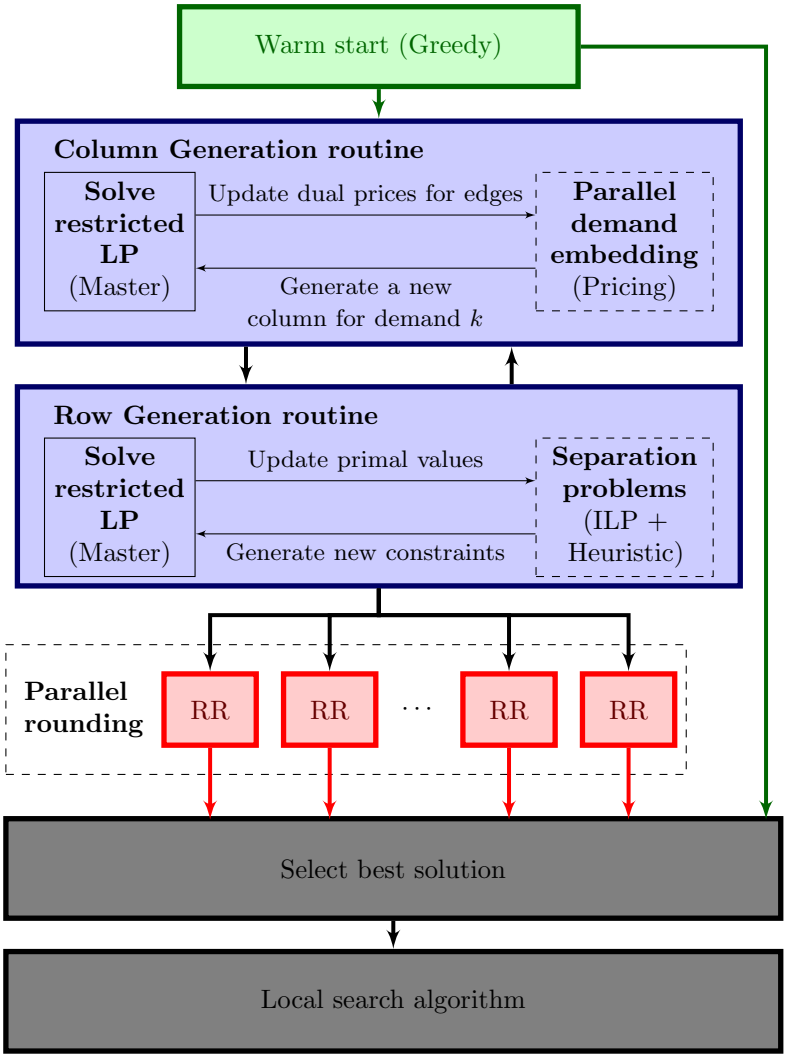


Figure 7: *FlexE-CG* (*allCuts*) algorithm.

stop due to a timeout, we still run the rounding algorithm one time to obtain an integer solution.

# 5 Polyhedral analysis

We now study the theoretical efficiency of the proposed inequalities by analyzing the polytope of the extended formulation. We provide necessary and sufficient conditions under which the 1-slot inequalities and edge-cut inequalities are facet-defining, and sufficient conditions under which the ratio capacity edge-cut inequalities are facet-defining. To ease the readability of the proofs, we restrict the analysis to the projection of the formulation on a particular edge-cut.

We focus on the following integer linear program.

$$\sum_{e\in\delta(V')}\sum_{s\in S^e}\xi_{es}y_{es} \geq \lceil D^r(V')\rceil \tag{22}$$

$$\sum_{e\in\delta(V')}\sum_{s\in S^e}\frac{1}{\mu^e}\xi_{es}y_{es} \geq \lceil D(V')\rceil \tag{23}$$

$$\sum_{s\in S^e} y_{es} \leq 1 \qquad e\in\delta(V') \tag{24}$$

$$y_{es}\in\{0,1\} \qquad e\in\delta(V'), s\in S^e \tag{25}$$

This integer linear program has a polynomial number of variables, whereas the original model has a exponential number of variables, which provides a better readability of our facet proofs.

Let $P(V')$ be the convex hull of the solutions of the integer linear program given by (22)-(25), that is,

$$P(V') = \text{conv}\left(y\in\{0,1\}^{\sum_{e\in\delta(V')}|S^e|} | y \text{ satisfy (22)-(24)}\right)$$

The vector $y^S$ is called the incidence vector associated with the solution $S$, i.e., $y^S = (y^S_{es})_{e\in\delta(V'),s\in S^e}$ where $y^S_{es}=1$ if $s\in S$ and $y^S_{es}=0$ otherwise.

## 5.1 Polytope dimension

Let $\bar{e}$ be the arc with the biggest capacity ($\arg\max_{e\in\delta(V')}\xi_{e\max}$), and $\tilde{e}$ be the arc with the biggest capacity ratio ($\arg\max_{e\in\delta(V')}\frac{1}{\mu^e}\xi_{e\max}$).

**Proposition 1** *Polytope $P(V')$ is full-dimensional if*

*a)* $\lceil D^r(V')\rceil \leq \sum_{e\in\delta(V')\setminus\bar{e}}\xi_{e\max}$

*b)* $\lceil D(V')\rceil \leq \sum_{e\in\delta(V')\setminus\tilde{e}}\frac{1}{\mu^e}\xi_{e\max}$

Proof ($\Rightarrow$)

- if $\lceil D^r(V')\rceil > \sum_{e\in\delta(V')\setminus\bar{e}}\xi_{e\max}$ then $\sum_{s\in S_e} y_{\bar{e}s}=1$
- if $\lceil D(V')\rceil > \sum_{e\in\delta(V')\setminus\tilde{e}}\frac{1}{\mu^e}\xi_{e\max}$ then $\sum_{s\in S_e} y_{\tilde{e}s}=1$

thus the polytope cannot be full dimensional.

($\Leftarrow$) we consider that a) and b) are true. We suppose that $P(V')$ is contained in a hyperplane defined by the linear equation

$$ay = \alpha \tag{26}$$

where $a = (a_{es}, e \in \delta(V'), s \in S_e) \in \mathbb{R}^{\sum_{e\in\delta(V')}|S_e|}$ and $\alpha \in \mathbb{R}$. We show that $a = 0$ and $P(V')$ cannot be included in the hyperplane (26), since the polytope is not empty.

We consider the solutions $\bar{S}_{e,s}$ for all $e \in \delta(V')$ and $s \in S_e$ such that for each arc $e' \in \delta(V') \setminus \{e\}$ the slot configuration is $s^{e'}_{\max}$ and $s$ on the arc $e$. Furthermore, we consider the solution $\bar{S}_{e,0}$ for all $e \in \delta(V')$ such that for each arc $e' \in \delta(V') \setminus \{e\}$ the slot configuration is $s^{e'}_{\max}$ and no slot configuration on the arc $e$. These solutions are clearly valid for the polytope $P(V')$.

By considering the solutions $\bar{S}_{e,s}$ and $\bar{S}_{e,0}$ for all $e \in \delta(V')$ and $s \in S_e$ since $ay^{\bar{S}_{e,s}} = ay^{\bar{S}_{e,0}}$, we deduce that $a_{es} = 0$ for all $e \in \delta(V')$ and $s \in S_e$. As there exists a solution, we can conclude that $a = 0$ and hence $P(V')$ is full dimensional.

In the following, we consider that $a$) and $b$) from the previous proposition is true and thus $P(V')$ is full dimensional. Indeed, if it is not the case we can reduce the possible slot configuration on some arcs.

## 5.2 Inequality (24) facet proof

**Proposition 2** *Inequality* (24) *associated with* $\bar{e} \in \delta(V')$ *defines a facet of* $P(V')$.

Proof Let us denote by $ay \leq \alpha$ the inequality (24), associated with $\bar{e} \in \delta(V')$. Let $by \leq \beta$ be a facet-defining inequality of $P(V')$ such that

$$\{y \in P(V') : ay = \alpha\} \subseteq \{y \in P(V') : by = \beta\}.$$

We show that $b = \rho a$ for some $\rho \in \mathbb{R}$.

We consider the solutions $\bar{S}_{e,s}$ for all $e \in \delta(V') \setminus \{\bar{e}\}$ and $s \in S_e$ such that for each arc $e' \in \delta(V') \setminus \{e\}$ the slot configuration is $s^{e'}_{\max}$ and $s$ on the arc $e$. Furthermore, we consider the solution $\bar{S}_{e,0}$ for all $e \in \delta(V') \setminus \{\bar{e}\}$ such that for each arc $e' \in \delta(V') \setminus \{e\}$ the slot configuration is $s^{e'}_{\max}$ and no slot configuration on the arc $e$. Remark that the incidence vectors of $\bar{S}_{e,s}$ and $\bar{S}_{e,0}$ satisfy inequality (24) with equality.

Since $ay^{\bar{S}_{e,s}} = ay^{\bar{S}_{e,0}}$, then $by^{\bar{S}_{e,s}} = by^{\bar{S}_{e,0}}$, implying that $b_{es} = 0$, for $e \in \delta(V') \setminus \{\bar{e}\}$ and $s \in S_e$.

Let $\bar{S}_{\bar{e},s}$ be a solution for all $s \in S_{\bar{e}}$ such that for each arc $e' \in \delta(V') \setminus \{\bar{e}\}$ the slot configuration is $s^{e'}_{\max}$ and $s$ on the arc $\bar{e}$. The incidence vectors of the solutions $\bar{S}_{\bar{e},s}$ verify inequality (24) with equality. Since $ay^{\bar{S}_{\bar{e},s_1}} = ay^{\bar{S}_{\bar{e},s_2}}$, $by^{\bar{S}_{\bar{e},s_1}} = by^{\bar{S}_{\bar{e},s_2}}$, implying that $b_{\bar{e},s_1} = b_{\bar{e},s_2}$ where $s_1, s_2 \in S_{\bar{e}}$. We set $b_{\bar{e},s_1} = \rho$, then $b_{\bar{e},s} = \rho$ for $s \in S_{\bar{e}}$.

Thus, there exists $\rho$ such that $b = \rho a$ for some $\rho \in \mathbb{R}$ and the proof is ended.

## 5.3 Ratio flow edge-cut inequalities facet proof

**Proposition 3** *Inequality* (8) *defines a facet of* $P(V')$ *only if*

$$\xi_{eS_<(\xi_{e\max})} \leq \lceil D^r(V') \rceil, \forall e \in \delta(V').$$

Proof Considering that there exists an arc $\bar{e} \in \delta(V')$ such that $\xi_{\bar{e}S_<(\xi_{\bar{e}\max})} > \lceil D^r(V') \rceil$. In this case, the last slot configuration of $\bar{e}$ will never be used, and we can deduce the following valid inequality

$$\sum_{e\in\delta(V')\setminus\{\bar{e}\}} \sum_{s\in S^e} \xi_{es} y_{es} + \sum_{s\in S^{\bar{e}}\setminus\{s^{\bar{e}}_{\max}\}} \xi_{\bar{e}s} y_{\bar{e}s} \geq \lceil D^r(V') \rceil. \tag{27}$$

This inequality dominates the inequality (8).

## 5.4 Ratio capacity edge-cut inequalities facet proof

**Proposition 4** *Inequality* (17) *defines a facet of* $P(V')$ *only if*

$$\frac{1}{\mu^e} \xi_{eS_<(\xi_{e\max})} \leq \lceil D(V') \rceil, \forall e \in \delta(V').$$

Proof Similar to the previous one.

## 5.5 1-slot inequalities facet proof

The 1-slot inequality associated with the 1-slot set $S^{\mathbb{1}}_{V'}$ is said maximal if and only if there does not exist $\bar{e} \in \delta(V')$ such that

$$\xi_{\bar{e}S^{\bar{e}}_{\leq}(\xi^{\nu\kappa}_{V'}(\bar{e}))} + \sum_{e \in \delta(V') \setminus \{\bar{e}\}} \xi_{eS^{e}_{<}(\xi^{\nu\kappa}_{V'}(e))} < \lceil D^r(V') \rceil,$$

or

$$\sum_{e \in \delta(V') \setminus \{\bar{e}\}} \frac{1}{\mu^e} \xi_{eS^{e}_{<}(\xi^{\nu\kappa}_{V'}(e))} + \frac{1}{\mu^{\bar{e}}} \xi_{\bar{e}S^{\bar{e}}_{\leq}(\xi^{\nu\kappa}_{V'}(\bar{e}))} < \lceil D(V') \rceil$$

where $\xi^{\mathbb{1}}_{V'}(e)$ is the size of the smallest slot configuration of arc $e$ for the 1-slot set.

**Proposition 5** *Inequality* (17) *associated with the 1-slot set* $S^{\mathbb{1}}_{V'}$ *defines a facet of* $P(V')$ *if and only if*

*a)* $S^{\mathbb{1}}_{V'}$ *is maximal,*

*b)* $\forall \bar{e} \in \delta(V')$ *there exists an edge* $e' \in \delta(V') \setminus \{\bar{e}\}$ *such that*

$$\sum_{e \in \delta(V') \setminus \{\bar{e}, e'\}} \xi_{eS^{e}_{<}(\xi^{1}_{V'}(e))} + \xi_{e' \max} \geq \lceil D^r(V') \rceil,$$

*or*

$$\sum_{e \in \delta(V') \setminus \{\bar{e}, e'\}} \frac{1}{\mu^e} \xi_{eS^{e}_{<}(\xi^{1}_{V'}(e))} + \frac{1}{\mu^{e'}} \xi_{e' \max} \geq \lceil D(V') \rceil.$$

Proof ($\Rightarrow$) a) If $S^{\mathbb{1}}_{V'}$ is not maximal then there exists an edge $\bar{e} \in \delta(V')$ such that

$$\sum_{e \in \delta(V') \setminus \{\bar{e}\}} \xi_{eS^{e}_{<}(\xi^{1}_{V'}(e))} + \xi_{\bar{e}S^{\bar{e}}_{\leq}(\xi^{1}_{V'}(\bar{e}))} < \lceil D^r(V') \rceil,$$

or

$$\sum_{e \in \delta(V') \setminus \{\bar{e}\}} \frac{1}{\mu^e} \xi_{eS^{e}_{<}(\xi^{1}_{V'}(e))} + \frac{1}{\mu^{\bar{e}}} \xi_{\bar{e}S^{\bar{e}}_{\leq}(\xi^{1}_{V'}(\bar{e}))} < \lceil D(V') \rceil$$

We can deduce that the inequality

$$\sum_{e \in \delta(V') \setminus \{\bar{e}\}} \sum_{s \in S^{e}_{\geq}(\xi^{1}_{V'}(e))} y_{es} + \sum_{s \in S^{\bar{e}}_{>}(\xi^{1}_{V'}(\bar{e}))} y_{\bar{e}s} \geq 1 \tag{28}$$

is valid and the inequality (17) associated with $S^{\mathbb{1}}_{V'}$ is linear combination of (28) and the inequality $y_{\bar{e}\xi^{1}_{V'}(\bar{e})} \geq 0$.

b) If there exists $\bar{e} \in \delta(V')$ such that for all edges $e' \in \delta(V') \setminus \{\bar{e}\}$, either

$$\sum_{e \in \delta(V') \setminus \{\bar{e}, e'\}} \xi_{eS^{e}_{<}(\xi^{1}_{V'}(e))} + \xi_{e' \max} < \lceil D^r(V') \rceil,$$

or

$$\sum_{e \in \delta(V') \setminus \{\bar{e}, e'\}} \frac{1}{\mu^e} \xi_{eS^{e}_{<}(\xi^{1}_{V'}(e))} + \frac{1}{\mu^{e'}} \xi_{e' \max} < \lceil D(V') \rceil,$$

then we can deduce

$$\sum_{e \in \delta(V') \setminus \{\bar{e}, e'\}} \sum_{s \in S^{e}_{\geq}(\xi^{1}_{V'}(e))} y_{es} + \sum_{s \in S^{\bar{e}}} y_{\bar{e}s} \geq 1 \ \forall e' \in \delta(V') \setminus \{\bar{e}\} \tag{29}$$

are valid. Indeed, for each slot configuration on $e'$, it is necessary to select at least one slot configuration of $S^{e}_{\geq}(\xi^{\mathbb{1}}_{V'}(e))$ for $e \in \delta(V') \setminus \{\bar{e}, e'\}$ or one slot configuration of $S^{\bar{e}}$. By summing (29) for each $\forall e' \in \delta(V') \setminus \{\bar{e}\}$,

the 1-slot inequality associated with a 1-slot set $S^{\mathbb{1}}$ and $|\delta(V')|-2$ the trivial inequality $\sum_{s\in S^{\bar{e}}_{\geq}(\xi^{\mathbb{1}}_{V'}(\bar{e}))} y_{\bar{e}s} \geq 0$ and dividing the resulting inequality by $|\delta(V')|-1$ and rounding up the right-and-side we obtain the following valid inequality

$$\sum_{e\in\delta(V')\setminus\{\bar{e}\}} \sum_{s\in S^{e}_{\geq}(\xi^{\mathbb{1}}_{V'}(e))} y_{es} + 2 \sum_{s\in S^{\bar{e}}_{\geq}(\xi^{\mathbb{1}}_{V'}(\bar{e}))} y_{\bar{e}s} + \sum_{s\notin S^{\bar{e}}_{\geq}(\xi^{\mathbb{1}}_{V'}(\bar{e}))} y_{\bar{e}s} \geq 2 \tag{30}$$

The 1-slot inequality associated with a 1-slot set $S^{\mathbb{1}}$ is a linear combination of the (30) and the inequality

$$-\sum_{s\in S^{\bar{e}}} y_{\bar{e}s} \geq -1$$

and thus cannot define a facet of $P(V')$.

($\Rightarrow$) Let us denote by $ay \geq \alpha$ the inequality (17), associated with the function $S^{\delta(V')}$. Let $by \geq \beta$ be a facet-defining inequality of $P(V')$ such that

$$\{y \in P(V') : ay = \alpha\} \subseteq \{y \in P(V') : by = \beta\}.$$

We show that $b = \rho a$ for some $\rho \in \mathbb{R}$.

For each $e \in \delta(V')$ and $s \in S^{e}_{\geq}(\xi^{\mathbb{1}}_{V'}(e))$, we denote by $\bar{S}_{es}$ a solution where the configuration slot $s^{\bar{e}}_{S^{\bar{e}}_{\leq}(\xi^{\mathbb{1}}_{V'}(\bar{e}))}$ is used for each $\bar{e} \in \delta(V')\setminus\{e\}$ and the configuration slot $s$ is used for the edge $e$. Remark that these solutions are valid since $S^{\mathbb{1}}$ is maximal.

For each $e \in \delta(V')$, we denote by $\bar{S}_{ee'\min}$ a solution where the slot configuration $s^{\bar{e}}_{S^{\bar{e}}_{\leq}(\xi^{\mathbb{1}}_{V'}(\bar{e}))}$ is used for each $\bar{e} \in \delta(V')\setminus\{e,e'\}$, and the slot configuration $s^{e}_{\max}$ is used for the arc $e' \in \delta(V')\setminus\{e\}$ and no slot configuration is used for the edge $e$. By the condition b) there exists $e' \in \delta(V')\setminus\{e\}$ such that $\bar{S}_{ee'\min}$ is valid.

For each $e \in \delta(V')$ the incidence vector of the solutions $\bar{S}_{ee'\min}$ satisfies the inequality (17) with equality. Since $ay^{\bar{S}_{ee'\min}} = ay^{\bar{S}_{e's'}}$, then $by^{\bar{S}_{ee'\min}} = by^{\bar{S}_{e's'}}$, where $s' \in S^{e'}_{\geq}(\xi^{\mathbb{1}}_{V'}(e'))$, which implies that $b_{es} = 0$ for each $s \in S^{e}_{<}(\xi^{\mathbb{1}}_{V'}(e))$ since $b_{e's'} = \rho$ for all $e' \in \delta(V')$ and for all $s' \in S^{e'}_{\geq}(\xi^{\mathbb{1}}_{V'}(e'))$.

For each $e_1, e_2 \in \delta(V')$, $s_1 \in S^{e_1}_{\geq}\xi^{\mathbb{1}}_{V'}(e_1)$ and $s_2 \in S^{e_2}_{\geq}(\xi^{\mathbb{1}}_{V'}(e_2))$, the incidence vectors of the solutions $\bar{S}_{e_1s_1}$ and $\bar{S}_{e_2s_2}$ satisfy the inequality (17) with equality. Since $ay^{\bar{S}_{e_1s_1}} = ay^{\bar{S}_{e_2s_2}}$, then $by^{\bar{S}_{e_1s_1}} = by^{\bar{S}_{e_2s_2}}$, which implies that $b_{e_1s_1} = b_{e_2s_2}$. We set $b_{e_1s1} = \rho$. Thus, $b_{es} = \rho$ for all $e \in \delta(V')$ and for all $s \in S^{e}_{\geq}(\xi^{\mathbb{1}}_{V'}(e))$.

# 6 Numerical results

Before evaluating our algorithms, we present the instances we used. Then, we extensively compare, on small instances, the efficiency of the various inequalities presented in Section 4 to improve the lower bound. Finally, we show the pros and cons of each algorithm on small, middle and large-scale instances in terms of optimality gap and execution times.

## 6.1 IP-RAN scenarios

We run our algorithms on realistic IP-RAN network topologies of various sizes, such as the one pictured in Figure 3. These networks are composed of a core network which connects multiple domains together. Each domain is composed of an access network connected to an aggregation network linked to the core. The core network is a random mesh graph, the aggregation networks are rings with probabilistic shortcuts and the nodes of the access networks (CSG nodes) are connected to the aggregation network in single or dual-homing. Access and core link capacities are set to 10Gbps while core links capacities are chosen at random between 40 and 200 Gbps. Services are generated at random between two CSG nodes (i.e., access nodes) or between a CSG node and the EPC (Evolved Packet Core), situated in the core network. They require between 100Mbps and 1Gbps amount of bandwidth. We consider three different sizes of instances: small (50 nodes, 60 links

and 60 demands), middle (1250 nodes, 1600 links and 300 demands) and large (5000 nodes, 6000 links and 600 demands). For each size, we generate 3 sets of 10 instances: one set of IP-RAN instances where CR services represent 80% of the traffic; one set with 100% CR services; and one set with 100% NC services. These different traffic scenarios allow us to see the impact between inequalities and their ratio variants.

## 6.2 Evaluation of inequalities

We start by analyzing the strength of each valid inequality described in Section 4. We consider eight scenarios:

- the *allCuts* procedure where we separate all inequalities to reinforce the master problem;
- all inequalities on their own (i.e., cover, ratio flow edge-cuts, ratio capacity edge-cuts and 1-slot inequalities with both ratio flow and ratio capacity version);
- and two combination of inequalities: *all flow* which is composed of the cover inequalities and the ratio flow version of the 1-slot inequalities and edge-cuts, and *all capacity* which is composed of the cover inequalities, ratio capacity edge-cuts and the ratio capacity version of the 1-slot inequalities.

The goal is to see the effectiveness of inequalities in improving the lower bound. We limit the evaluation of inequalities to small instances due to the excessive running time required for the experiment.

**Lower bound quality.** We show, in Figure 8, the lower bound provided by our valid inequalities compared to the lower bound of the basic extended formulation using column generation (i.e., without additional cuts). We observe that edge-cut inequalities and 1-slot inequalities and their ratio variant provide, in average, around 6% improvement on the lower bound. On their own, cover inequalities can improve the lower bound by almost 7.5%. Separating different inequalities together can further improve the lower bound to almost 9% while 50% of the instances see an improvement between 7 and 10%. The *allCuts* procedure only shows a 0.1 to 0.2% improvement compared to the *all ratio* and *all capacity* scenarios, respectively.

**Solution quality.** Figure 9 shows the distribution on all instances of the ratio between the objective value obtained with a given set of cuts over the one obtained by *FlexE-CG*. We observe that inequalities barely impact the quality of the solution obtained. Edge-cut inequalities, 1-slot inequalities and their variants have, in average, almost no impact, giving solutions close to the one given by the basic *FlexE-CG* algorithm. This can be explained by the nature of the inequalities: they do not contain path variables and thus barely impact on the number of generated columns. But cover inequalities contain the path variables, and they can improve the solutions by about 0.5%. Finally, combining all inequalities barely change the situation, with a slight improvement over the separation only cover inequalities. Since we use a randomized rounding approach to obtain the integer solution from our column generation algorithm, using inequalities can lead to worse solutions, with an increase of the objective function by up to 3% in some cases.

**Computational time.** Figure 10 shows the computational time for each type of inequalities compared to *FlexE-CG* (i.e., basic version without inequalities). Over all classes, cover inequalities takes the longest to separate with an average of 14.9 s. Since we only consider a fixed subset of edge-cuts in the network for edge-cut, 1-slot and their variants, their separation problems are faster to execute: 1-slot takes an average of 2.1 s and edge-cut takes an average of 1.0 s to separate. Separating all inequalities together leads to a slight increase, with an average of 20 s. We also see that in IP-RAN scenarios with convergence ratio, the algorithms need more time to converge, due to the mix of both types of demands.

Even though separating all inequalities does not improve the quality of the solution, it provides a good improvement of the lower bound, which we can use to evaluate the optimality gap on larges instances where the ILP cannot be solved using CPLEX (see later in Section 6.3). The impact on the computational time is reasonable, with an increase up to 15%, at least on small instances.

## 6.3 Lower bounds comparison

We now compare, in Figure 11, the bound provided by *FlexE-CG*(allCuts) and the ones by *FlexE-CG*. We also compare them with the bounds provided by the compact formulation solved using CPLEX, denoted ILP, on small instances.

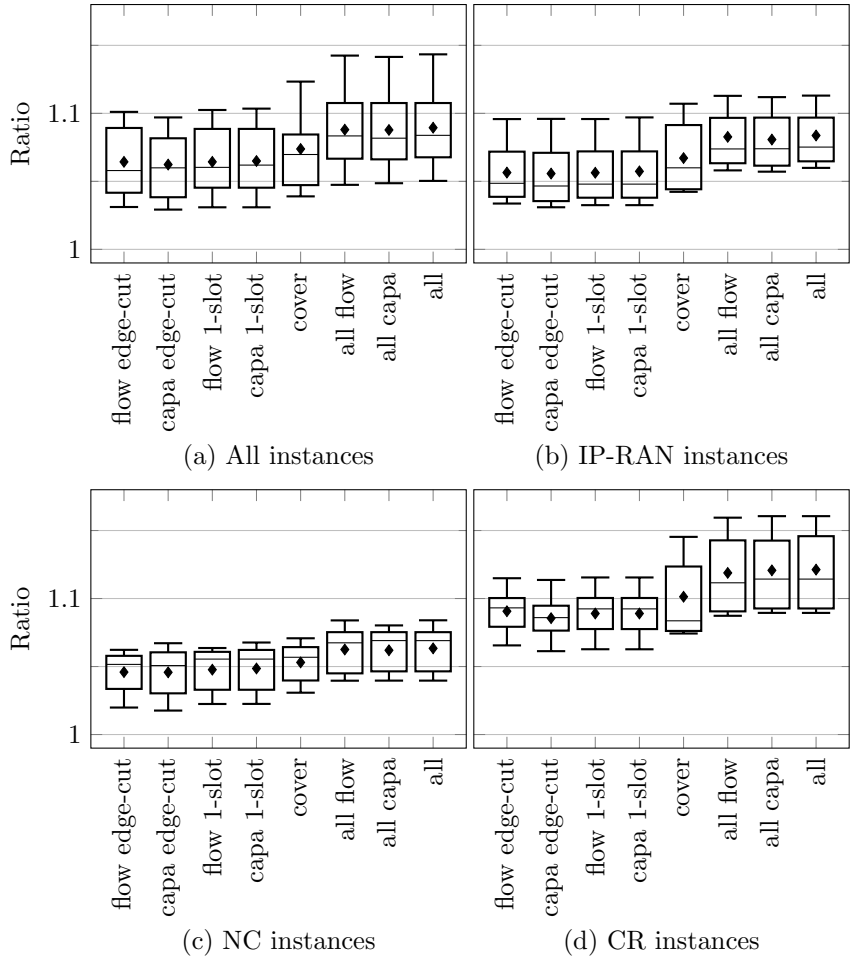


Figure 8: Comparisons of the lower bound ratio for all traffic scenarios and each set of inequalities on small instances. The box plots show the distribution on all instances of the ratio between the bounds obtained with the given cut over the one obtain with *FlexE-CG* (higher is better). Whiskers represent the first and last deciles, the box represents the first quartile, the median and the third quartile, and the marker represents the mean.

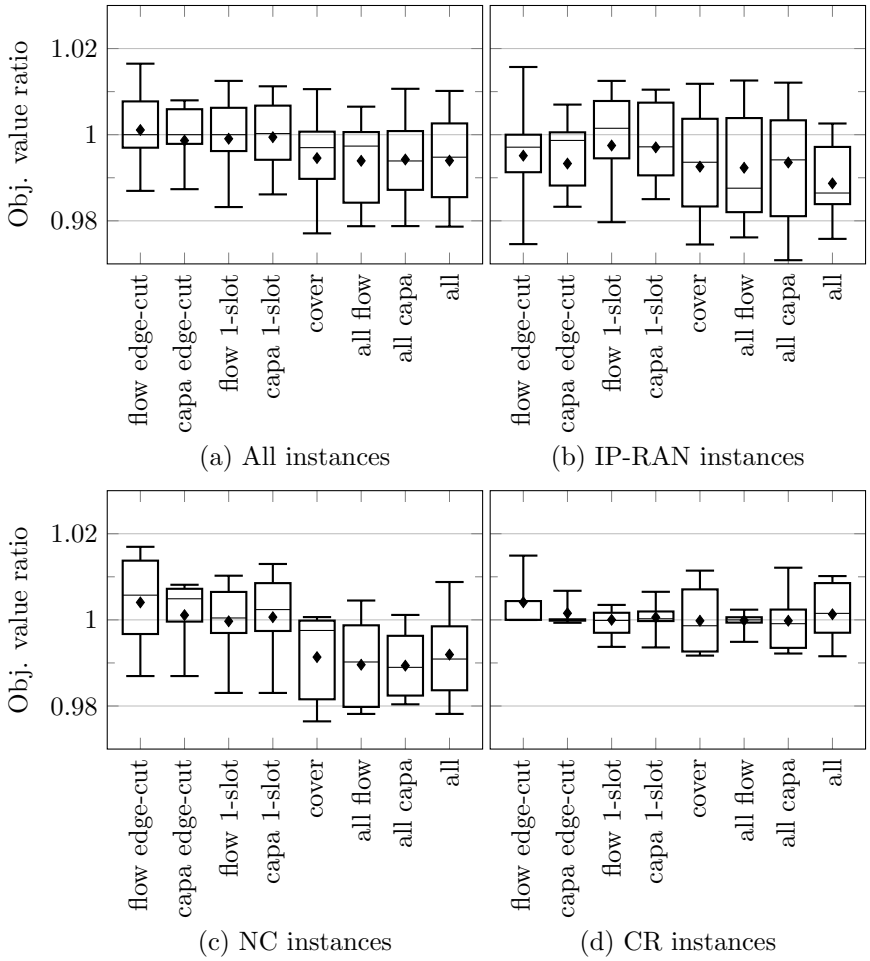


Figure 9: Comparisons of the objective value when using different inequalities on small instances. The box plots show the distribution on all instances of the ratio between the objective value obtained with the given cut over the one obtained by *FlexE-CG* (lower is better). Whiskers represent the first and last deciles, the box represents the first quartile, the median and the third quartile, and the marker represents the mean.

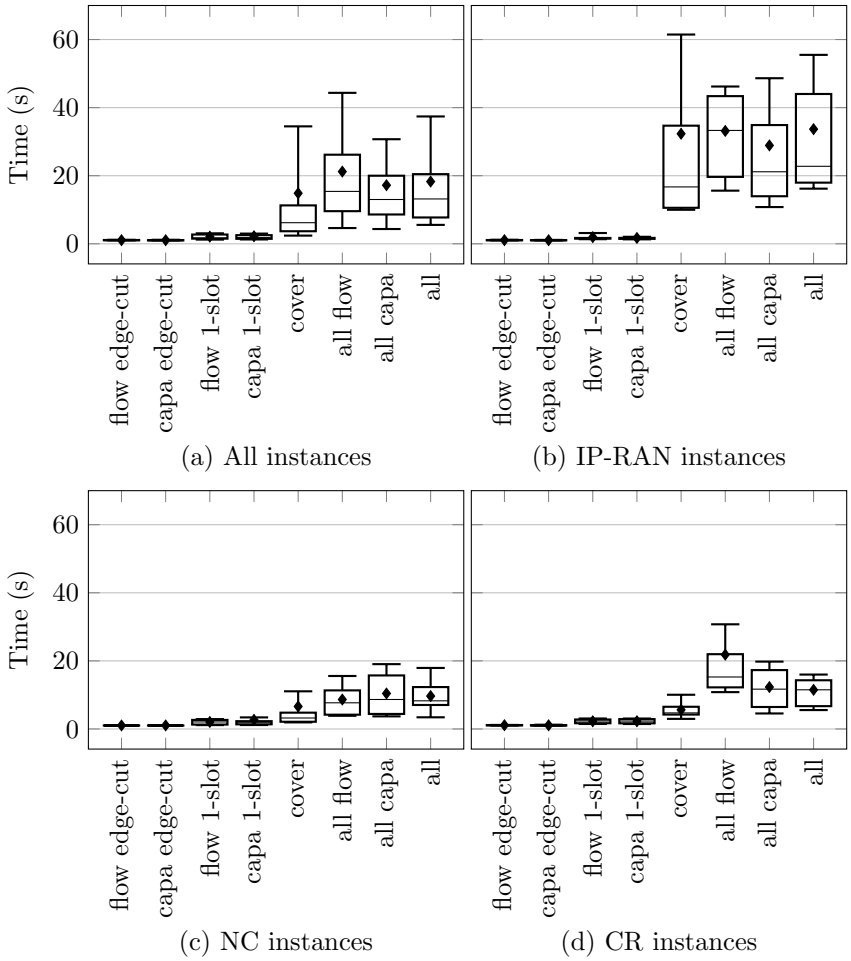


Figure 10: Comparisons of the computational time when using different inequalities on small instances. The box plots show the distribution on all instances of the ratio between the computational time of using a given cut over the computational time of *FlexE-CG* (lower is better). Whiskers represent the first and last deciles, the box represents the first quartile, the median and the third quartile, and the marker represents the mean.

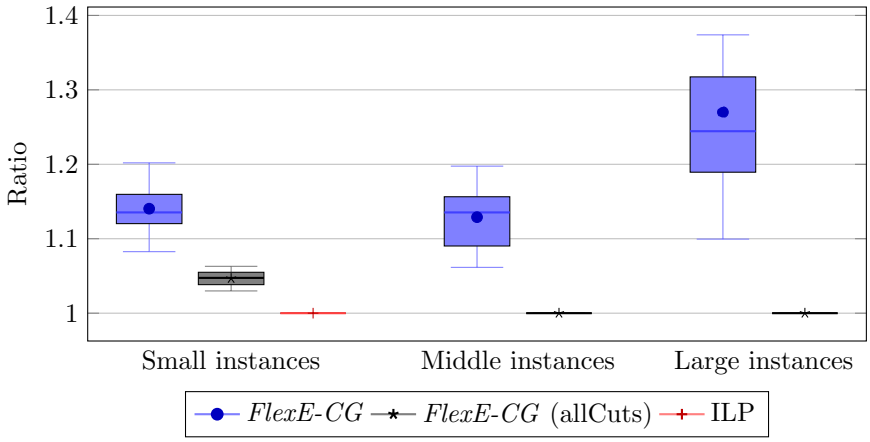


Figure 11: Comparisons of the bounds obtained by *FlexE-CG*(allCuts), the *FlexE-CG*, and ILP. The box plots show the distribution on all instances of the ratio between the bounds obtained by the given algorithm and the best known bound. The best known bound is given by ILP on small instances and is given by *FlexE-CG*(allCuts) on middle and large instances. Whiskers represent the first and last decile, the box represents the first quartile, the median and the third quartile, and the marker represents the mean.

*FlexE-CG*(allCuts) provides good lower bounds and can be used to assess the quality of an algorithm: on small instances, the bounds are in average within 3% of the ILP ones, while *FlexE-CG* one are within 13%. On middle and large instances, the improvement is even greater with 12% and 27%, respectively. It can even reach an 85% improvement on large instances.

## 6.4 Algorithm comparison

We now compare the algorithms proposed in Section 3.2:

- Greedy: the greedy algorithm, described in Algorithm 1;
- ILP: the compact formulation solved using CPLEX 12.6;
- *FlexE-CG*: our column generation based heuristic without inequalities separation, described in Section 3.3.;
- *FlexE-CG*(allCuts): our column generation based heuristic extended with the *allCuts* separation procedure described in Section 4.4.

For all algorithm, we imposed a time limit of one hour.

**Gap to the best bound.** We first compare the results of the solutions obtained with the best lower bound available for each instance. On small instances, this gap is provided by the best lower bound obtained by CPLEX while solving the compact formulation: if the one-hour time limit is reached, we use the best bound found during the branch-and-bound; otherwise, we use the optimal value. On middle and large instances, we use the lower bound provided by *FlexE-CG*(allCuts) at the end of the column generation step. Let $z_{\text{LP}}$ be the best lower bound found for an instance, and $\tilde{z}$ the value of the algorithm solution, the gap is given by $1 - \frac{\tilde{z}}{z_{\text{LP}}}$.

Figure 12 shows the gap to the best lower bound of all four algorithms. Over all instances, *FlexE-CG* provides solutions that have a lower gap than the ones of the Greedy algorithm. On average, they have an average gap of 6% while Greedy's solutions have an average gap of 16%, and we observe this trend for all size classes.

On small instances, the ILP cannot always provide optimal solution (or at least proves optimality) in the imposed time limit. *FlexE-CG* solutions are, in 90% of the instances, below a 10% gap while 80% of the

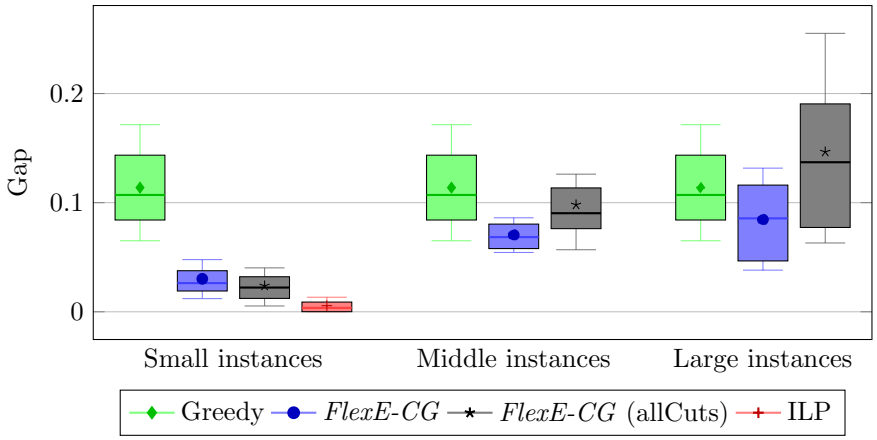


Figure 12: Gap of the solution to the best lower bound. Whiskers represent the first and last deciles, the box represents the first quartile, the median and the third quartile, and the marker represents the mean.

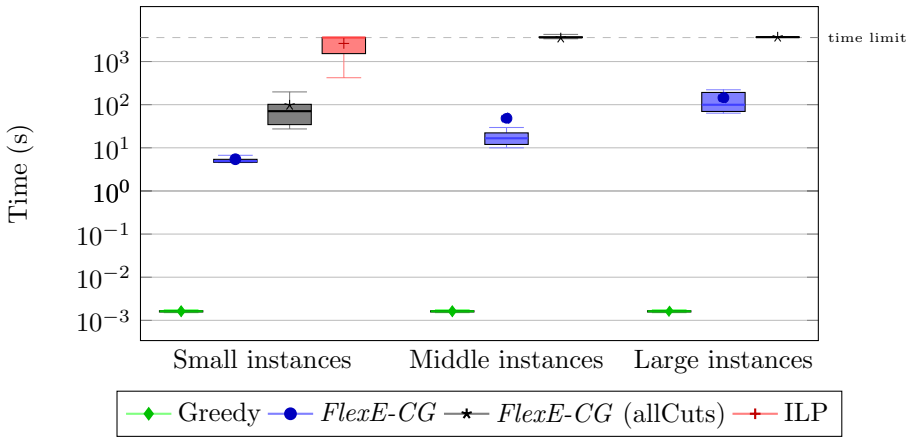


Figure 13: Comparisons of the running time of all four algorithms. Whiskers represent the first and last deciles, the box represents the first quartile, the median, and the third quartile, and the marker represents the mean.

greedy solutions are between 5% and 20%. Finally, *FlexE-CG*(allCuts) gives slightly better solutions than *FlexE-CG*.

When looking at middle and large instances, *FlexE-CG*(allCuts) behaves poorly compared to *FlexE-CG* and, on some large instances, worse than Greedy. This can be explained by the one-hour time limit. Indeed, with enough time to converge, like for small instances, *FlexE-CG*(allCuts) can find better solutions than *FlexE-CG*. However, without sufficient time, the cut generation part of the algorithm hurts the rounding phase by limiting its available running time, which leads to the worst solutions.

**Computation time.** Figure 13 shows the running time of all four algorithms. As previously mentioned, all algorithms are limited to one hour. Greedy is the fastest algorithm with less than 10 ms on small instances, less than 100 ms on middle instances and less than 200 ms on large instances. *FlexE-CG* is the second-fastest algorithm with an average of 5 s on small instances, 50 s on middle instances and 150 s on large instances. And as previously stated, the ILP mostly reaches the time limit of one hour on small instances. *FlexE-CG*(allCuts) has the second-worst performance with an average 100 s to solve small instances. Unfortunately, it reaches the one-hour time limit on all middle and large instances. We point out that creation and the deployment of a slice are time-consuming operations as it is required to reserve the resources, deploy the corresponding IP

addresses and let the different running protocols, such as IGP, converge. is Normally, several minutes to hours are needed to conclude these operations, differently from single service provisioning in classical networks, where few milliseconds only are a strict time computation requirement. For this reason, the running time of the algorithm, which is in the order of tens-hundreds seconds is not critical.

# 7 Conclusions

In this paper, we presented the Routing and Slot Allocation (RSA) problem for 5G hard slicing. We modelled the problem using mathematical programming and proposed an extended formulation that we solve using column generation. We derived a column generation heuristic, *FlexE-CG*, that combines a greedy and a local search algorithms. We introduced new cutting planes for the problem to reinforce the extended formulation and improve the lower bound. We proposed three families of inequalities: *cover* inequalities, *edge-cuts* inequalities and *1-slot* inequalities. And we provided necessary and sufficient conditions under which the edge-cut and 1-slot inequalities are facet-defining and sufficient conditions under which the edge-cut inequalities are facet-defining.

Using realistic IP-RAN network instances, we showed that these inequalities (which give the *FlexE-CG*(allCuts) when added to *FlexE-CG*) are efficient in improving the lower bound by up to 25% compared to the cut-less version of the extended formulation. However, *FlexE-CG*(allCuts) suffers from a high running time as it reaches the one-hour time limit and from poor solution quality with an average of 15% optimality gap. Our proposed algorithm, *FlexE-CG*, provides solutions within a 10% gap in a couple of minutes on large instances, making it feasible within the context of 5G networks offline planning of a single slice..

# References


[1] D. Applegate and E. Cohen. "Making Intra-Domain Routing Robust to Changing and Uncertain Traffic Demands: Understanding Fundamental Tradeoffs". In: *ACM SIGCOMM 2003*. Karlsruhe, Germany: Association for Computing Machinery, 2003, pp. 313–324.

[2] Mohamad Khattar Awad, Yousef Rafique, and Rym A. M'Hallah. "Energy-aware routing for software-defined networks with discrete link rates: A benders decomposition-based heuristic approach". In: *Sustainable Computing: Informatics and Systems* (2017).

[3] D. Awduche et al. *RSVP-TE: Extensions to RSVP for LSP Tunnels*. 2001.

[4] Y. Azar et al. "Optimal Oblivious Routing in Polynomial Time". In: *Proc. ACM STOC*. 2003, pp. 383–388.

[5] Egon Balas. "Facets of the knapsack polytope". In: *Mathematical programming* 8.1 (1975), pp. 146–164.

[6] Pietro Belotti et al. "Multi-layer MPLS network design: The impact of statistical multiplexing". In: *Computer Networks* 52.6 (2008), pp. 1291–1307.

[7] W. Ben-Ameur and H. Kerivin. "Networks New Economical Virtual Private". In: *Commun. ACM* 46.6 (June 2003), pp. 69–73.

[8] W. Ben-Ameur and H. Kerivin. "Routing of Uncertain Traffic Demands". In: *Optimization and Engineering* 6.3 (Sept. 2005), pp. 283–313.

[9] Walid Ben-Ameur, Adam Ouorou, and Mateusz Zotkiewicz. "Robust routing in communication networks". In: *Progress in combinatorial optimization* (2011), pp. 353–390.

[10] Daniel Bienstock and Oktay Günlük. "Capacitated Network Design—Polyhedral Structure and Computation". en. In: *INFORMS Journal on Computing* 8.3 (Aug. 1996), pp. 243–259.

[11] A. Capone, G. Carello, and R. Matera. "Multi-Layer Network Design with Multicast Traffic and Statistical Multiplexing". In: *IEEE GLOBECOM 2007 - IEEE Global Telecommunications Conference*. 2007, pp. 2565–2570.

[12] C. Chekuri et al. “Hardness of robust network design”. In: *Networks* 50.1 (2007), pp. 50–54.

[13] Franck Chevalier, John Krzywicki, and Mike Pearson. *Optical Transport Network (OTN) And/Or Multi-Protocol Label Switching (MPLS)? That is the question. Juniper Whitepaper. 2012*. 2011.

[14] China Mobile Communications Corporation et al. *5G Service-Guaranteed Network Slicing White Paper*. Tech. rep. Huawei, Feb. 2017.

[15] Alysson M. Costa. “A survey on benders decomposition applied to fixed-charge network design problems”. en. In: *Computers & Operations Research* 32.6 (June 2005), pp. 1429–1450.

[16] Guy Desaulniers, Jacques Desrosiers, and Marius M Solomon. *Column generation*. Springer Science & Business Media, 2006.

[17] A. Destounis et al. “Slice-based column generation for network slicing”. In: *IEEE INFOCOM 2018 - Poster*. Apr. 2018, pp. 1–2.

[18] J. Dong et al. *A Framework for Enhanced Virtual Private Networks (VPN+) Services*. draft-ietf-teas-enhanced-vpn-05. 2020.

[19] C. Filsfils et al. *Segment Routing Architecture*. 2020.

[20] X. Foukas et al. “Network Slicing in 5G: Survey and Challenges”. In: *IEEE Communications Magazine* (May 2017).

[21] Antonio Frangioni and Bernard Gendron. “0–1 reformulations of the multicommodity capacitated network design problem”. In: *Discrete Applied Mathematics* 157.6 (2009), pp. 1229–1241.

[22] V. Gabrel, A. Knippel, and M. Minoux. “Exact solution of multicommodity network optimization problems with general step cost functions”. In: *Operations Research Letters* 25.1 (1999), pp. 15–23.

[23] Liang Geng et al. *Network Slicing Architecture*. Internet-Draft draft-geng-netslices-architecture-02. Work in Progress. Internet Engineering Task Force, July 2017. 27 pp.

[24] Nicolas Huin et al. “Hard-isolation for network slicing”. In: *IEEE INFOCOM (WORKSHOPS)*. IEEE. 2019, pp. 955–956.

[25] IBM. *ILOG CPLEX Solver*. 2022.

[26] A. Juttner et al. “Lagrange relaxation based method for the QoS routing problem”. In: *Proc. IEEE INFOCOM 2001*. Vol. 2. 2001, 859–868 vol.2.

[27] Eytan Modiano. “Traffic grooming in WDM networks”. In: *IEEE communications Magazine* 39.7 (2001), pp. 124–129.

[28] OIF. *Flex Ethernet 2.0 Implementation Agreement*. June 2018.

[29] M. Poss. “A comparison of routing sets for robust network design”. In: *Optimization Letters* 8.5 (June 2014), pp. 1619–1635.

[30] Christian Raack et al. “On cut-based inequalities for capacitated network design polyhedra”. In: *Networks* 57.2 (2011), pp. 141–156.

[31] Laurence A Wolsey and George L Nemhauser. *Integer and combinatorial optimization*. Vol. 55. John Wiley & Sons, 1999.

[32] Ying Xiao, Krishnaiyan Thulasiraman, and Guoliang Xue. “GEN-LARAC: A Generalized Approach to the Constrained Shortest Path Problem Under Multiple Additive Constraints”. In: *Algorithms and Computation*. Ed. by Xiaotie Deng and Ding-Zhu Du. Springer Berlin Heidelberg, 2005, pp. 92–105.