Two extended formulations for the Virtual Network Function Placement and Routing Problem

Ahlam Mouaci[a,b], Éric Gourdin[a], Ivana Ljubić[c], Nancy Perrot[a]

[a] *Orange Innovation, Paris, France*
[b] *Université Paris Dauphine, Paris, France*
[c] *ESSEC Business School, Cergy, France*

Abstract

Given a bi-directed graph modeling a telecommunication network, and a set of origin-destination pairs representing traffic requests (commodities) along with their associated Service Function Chains (SFCs), the Virtual Network Function Placement and Routing Problem (VNFPRP) aims to find, for each commodity, one latency-constrained routing path that visits the required Virtual Network Functions in a specific order. The function installation costs together with the node activation costs have to be minimized.

In this paper, we present two extended Mixed Integer Programming (MIP) formulations to model the VNFPRP. For each formulation we define the master problem, the pricing problem, the associated Lagrangian bound and a specific branching scheme, in order to derive an efficient Branch-and-Price algorithm. We also provide several families of valid inequalities to strengthen the LP-relaxation bounds. Computational results are reported comparing the performance of the two Branch-and-Price algorithms with a compact MIP formulation and its Branch-and-Benders-cut implementation on a set of SNDlib instances representing telecommunication networks.

Keywords: Networks, Column generation, Branch-and-Price, Network Function Virtualization, Service Functions Chaining.

1. Introduction

The Virtual Network Function Placement and Routing Problem (VNFPRP) is a topical problem in the design of 5G networks. In this problem, we are given a telecommunication network, a set of Virtual Network Functions (VNFs), which correspond to software that can be installed on-demand at network nodes. In addition, we are given a set of customer demands (also called traffic requests or commodities) asking for services like video streaming, virtual private network (VPN), or on-line gaming. Each customer demand has to be treated by a given subset of VNFs in a given order, defined by a Service Function Chain (SFC). SFC is a sequence of VNFs that should be traversed by a given service flow in a predefined order [26]. For example, the network administrator may specify a policy that all `http` traffic should follow the policy chain: "`firewall` $\rightarrow$ `IDS` $\rightarrow$ `proxy`" [20].

The infrastructure around VNFs requires a system that performs central orchestration and management of traffic requests, making sure that the traffic is routed in such a way that service

Email addresses: `ahlam.mouaci-magnouche@dauphine.psl.eu` (Ahlam Mouaci), `eric.gourdin@orange.com` (Éric Gourdin), `ivana.ljubic@essec.edu` (Ivana Ljubić), `nancy.perrot@orange.com` (Nancy Perrot)

function chaining constraint is respected. This is achieved with the Software Defined Networking (SDN) technology [6]. Thanks to VNFs and SDN, the telecommunication networks are becoming dynamic and programmable. Moreover, VNFs and SDN help facilitate the service management, minimize the service installation and management costs, and decrease deployment delays.

The VNFPRP studied in this paper consists of determining: a) the optimal installation of VNFs at the network nodes, and b) the routing of each traffic demand through a latency-constrained path satisfying the SFC constraints. The costs for activating network nodes and installing VNFs on them have to be minimized. Further constraints can be considered, such as node and VNF capacities, or conflicts between VNFs when installed at the same node.

In this paper, we consider the following assumptions:

- There is sufficient link capacity in the existing network to handle all traffic requests, due to the fact that we are dealing with the *tactical* planning rather than with *strategic* network design (in which the link capacities are designed according to anticipated traffic requests).
- The routing paths should be elementary (without circuits) to avoid the network loops.
- Each commodity requires the installation of at least one VNF (otherwise, without loss of generality, these commodities can be eliminated in a pre-processing phase). There is a (partial) order imposed between VNFs that need to be installed for each commodity.
- Multiple (heterogeneous or the same) VNFs can be installed at the same node in the network.
- A commodity can use the capacity of multiple copies of the same VNFs in the same node.

In this paper we deal with a variant of the VNFPRP, in which we relax the link capacity constraints based on the assumption cited before. Furthermore, allowing creation of only elementary paths, aims to minimise the arcs utilization, which is considered as an objective function in multiple earlier works. We consider also node and VNFs capacity constraints which were relaxed in our previous work [21], in order to reduce the resolution time. We notice also that adding these constraints in the model proposed in [21] will in any case change the characteristics of the proposed model, but will just generate new solutions more suited for the original version of problem. Also, this will allow us to compare both works and see which decomposition is better for this type of problems. The version of the problem considered in this paper includes most of the technical constraints encountered by operational teams in a real-world setting. As a result, its resolution is very challenging. To the best of our knowledge, the decomposition methods we propose have not been investigated before for the VNFPRP.

Our contribution. Previous papers related to the VNFPRP mainly rely on compact MIP formulations and heuristics (see, e.g., [1, 2, 3, 13, 15, 30]). These studies show that the computational performance of compact formulations is fairly limited, mainly because of their relatively weak LP-relaxation bounds. In this work, to enhance the capabilities of finding exact solutions for larger instances of practical relevance, we propose two *extended MIP formulations* for the VNFPRP. In both formulations, we consider an extended variable space admitting an exponential number of variables which allows us to develop tighter MIP reformulations.

The first reformulation (denoted as PF) was proposed in [25], where it was used for deriving a heuristic approach. In the current paper, we provide a branch-and-price algorithm to solve exactly this PF model. Therein, we separate the VNF placement problem, which is treated at the master level, from the routing problem, which is solved separately for each commodity in the pricing problem. The second extended formulation (denoted as DW) is an extended formulation obtained from an alternative Dantzig-Wolfe decomposition approach. The problem is decomposed per commodity in such a way that: the master problem ensures that exactly one path with its associated VNF installations is chosen for each commodity, and that node and VNFs capacity constraints are satisfied. The routing and VNF placement constraints associated with each commodity are managed in the pricing problems. In order to improve the LP-relaxation bounds of our formulations, new families of valid inequalities are also proposed. All these elements are combined in two efficient Branch-and-Price algorithms.

Closely related to the VNFPRP is a problem that we studied in [21], where the VNF installation costs are paid per commodity, whereas in this paper they are paid per VNF. Moreover, in [21], different capacity constraints were considered. For that problem variant, we proposed in [21] a compact flow-based formulation and a Branch-and-Benders-cut approach derived from projecting out flow variables. In the current work, we adapt the compact MIP formulation and the Branch-and-Benders-cut approach in order to cope with additional constraints and the modified objective function given by the VNFPRP. In our detailed computational study, the latter two approaches are compared against to the two Branch-and-Price algorithms.

The LP-relaxation bounds of the two extended formulations dominate the respective bounds of the compact formulation and its Benders reformulation. This is also confirmed by our empirical study which also shows that the strongest bounds among all four formulations are obtained with the model DW. Our study also reveals that the formulation DW suffers from a trade-off between the strength of the obtained bounds and the computational time needed for solving the pricing problem.

Outline of the paper. In Section 2, we introduce the main notation, provide the formal problem definition and give an overview of the related literature. In Sections 3 and 4, we present the two extended formulations, we discuss the associated pricing problems, detail the computation of the Lagrangian bound and present branching schemes. In Section 5, we provide a set of valid inequalities that aim to strengthen the LP-relaxation bounds. In Section 6 we provide implementation details of the Branch-and-Price (B&P) algorithms for each formulation. In Section 7, we discuss the obtained numerical results and we conclude the paper with some remarks and perspectives in Section 8. More detailed computational results are available in the Appendix.

2. Problem definition and related work

Notation. The telecommunication network is modeled as a bi-directed graph $G = (N, A)$. The set of all physical locations equipped with hardware devices allowing VNFs installation is denoted by N, and called the set of nodes. The set representing all directed links between nodes is called the set of arcs, and is denoted by A. At each node $u \in N$ at most c_u ($c_u \in \mathbb{N}$) VNFs can be installed, and an activation cost $\psi_u > 0$ has to be paid. The arc latency $l_{uv} > 0$ is defined for each arc $(u, v) \in A$.

Let F denotes the set of all virtual network functions. Each VNF $f \in F$ has a limited (bandwidth) capacity m_f to manage the traffic demand. For installing a single copy of f at a node $u \in N$, an installation cost $\psi_u^f > 0$ has to be paid. For the set of all traffic requests/commodities, denoted by C, each commodity $k \in C$ is characterized by: a source node, s_k; a destination node, d_k ($s_k \neq d_k$); a bandwidth $b_k > 0$; a maximum latency value $l_k > 0$; a subset of VNFs, $F^k \subseteq F$, and a set of incompatibility constraints $\mathcal{A}^k$ between VNFs. Recall that each commodity requires a specific Service Function Chain to handle its data packages in a specific (possibly partial) order. The fact that the VNF f has to be executed before VNF g in the path associated to the commodity k is expressed by $f \prec_k g$. Table 1 summarizes all the notation.

Sets		
N	:	Set of all nodes.
A	:	Set of all arcs.
C	:	Set of all commodities (traffic requests).
F	:	Set of all Virtual Network Functions.
$(F^k, \prec_k)$	:	(Partially) ordered set of VNFs associated with commodity k, $k \in C$.
$\mathcal{A}^k$	:	Set of pairs of VNFs which are in conflict for commodity k, $k \in C$.
Parameters		
m_f	:	Capacity of VNF f, $f \in F$.
c_u	:	Capacity of node u, $u \in N$.
l_{uv}	:	Latency of link (u, v), $(u, v) \in A$.
ψ_u^f	:	Installation cost of VNF f at node u, $f \in F$, $u \in N$.
ψ_u	:	Activation cost of node u, $u \in N$.
s_k	:	Source node for commodity k, $k \in C$.
d_k	:	Destination node for commodity k, $k \in C$.
l_k	:	Maximum latency of commodity k, $k \in C$.
b_k	:	Bandwidth of commodity k, $k \in C$.

Table 1: Main notation, parameters and sets.

Definition 1 (VNFPRP). *The Virtual Network Function Placement and Routing Problem consists of finding for each commodity $k \in C$, the placement of VNFs $f \in F^k$ at nodes, and the routing paths so that the sum of the VNF installation costs ψ_u^f plus the sum of node activation costs ψ_u is minimized. The following constraints have to be satisfied:*

- ***Node-capacity constraints:*** *each node $u \in N$ has an installation capacity c_u, which means that the number of VNFs installed at u should not exceed c_u.*
- ***VNF-capacity constraints:*** *each VNF $f \in F$ has a capacity m_f to manage the amount of data. The sum of bandwidths handled by one copy of f should be at most m_f.*
- ***Routing constraints:*** *the $s_k - d_k$ routing path associated with each commodity $k \in C$ should be elementary.*
- ***End-to-end latency constraints:*** *the sum of arc latencies belonging to the routing path of each commodity $k \in C$ should not exceed the given latency limit of l_k.*
- ***Installation constraints:*** *each VNF $f \in F^k$ required for commodity $k \in C$ should be installed at one of the nodes of the routing path.*
- ***Precedence constraints:*** *for each commodity $k \in C$, the VNFs composing its Service Function Chain should be traversed in the right order by the routing path.*

- ***Conflict constraints*** *(also called anti-affinity/incompatibility constraints): For each $k \in C$, two functions f and g which are in conflict (i.e., $\{f, g\} \in \mathcal{A}^k$) cannot be installed at the same node. These constraints are generally used to reduce the impact of an infrastructure failure [9]*

In our previous work, we have shown the following result:

Theorem 2.1 ([21]). *The VNFPRP is strongly NP-hard, even without latency, capacity and precedence constraints and with a single commodity.*

2.1. Related work

In the context of SDN and VNFs, the major decision problem consists of finding an optimal installation of VNFs, so that the given traffic requests can be routed within the given network infrastructure while respecting the SFC constraints [1, 14, 22, 23]. Many different variations of the problem have been studied in the recent literature. When it comes to the definition of the objective function, some authors consider non-linear objective functions, others are minimizing the utilization of the network elements (like the number of links involved), and others are minimizing the VNF installation costs. In the sequel, we provide an overview of the existing literature.

In [21], we have studied a variant of the VNFPRP, where the VNF installation costs are paid per commodity, instead of being paid per VNF (as in the current paper) and with slightly different capacity constraints. We proposed a compact flow-based MIP formulation to model the problem and provided theoretical results that allowed us to reformulate the problem using Benders decomposition. We also introduced additional valid inequalities and a MIP-based heuristic, the performance of which we studied in [25]. We combined all these ingredients in a Branch-and-Benders-Cut framework and tested it on two sets of (realistic and randomly generated) benchmark instances. The obtained results have shown that this algorithm outperforms the compact MIP formulation, and the automatic Branch-and-Benders-Cut implementation provided by Cplex.

Other problem variants. Different variants were proposed for modeling a simultaneous VNF placement and routing problem. Some articles study the problem in which only single VNF has to be installed per commodity. For example, Addis et al. [2], propose two alternative compact MILP formulations to deal with a VNFPR variant in which a single service is installed per each commodity (i.e., $|F_k| = 1$, for all $k \in C$). The authors compare the two formulations empirically and with respect to the strength of their LP relaxations. They also show that the two models can be generalized to handle multiple services (see also [11]). Bouet et al. [5] consider the problem of placing the virtualized Deep Packet Inspection in SDN networks. They propose a method based on genetic algorithms, that optimizes the cost of DPI engine deployment, minimizing their number, the global network load and the number of not analyzed flows, while considering arc capacity constraints. Moens & De Turck [23], propose a MIP model where both physical and virtual resources are allocated to the function chains, which is tested with very small instances using Cplex solver.

Several works assume a total order of VNFs and exploit this fact to model the problem as layered graph to deal with the chaining constraints. Sallam et al. [27] study the Shortest Path and Maximum Flow Problems under Service Function Chaining constraints for which virtual and not virtual network functions are considered. Authors solve the SFC-constrained shortest path problem by transforming the graph into a layered graph. Another layered graph formulation and the associated column generation approach for the VNFPRP with link capacities have been proposed in [16]. In contrast to our models in which partial ordering of VNFs is allowed, the formulation given in [16] is only valid when a total order for VNFs is provided. In a related study by Tomassilli et al. [31], a compact MIP model and a column generation approach are proposed for another variant of the problem in which latency, precedence, flow, link and node capacity constraints are considered.

In other related papers, different objective functions are tackled. For example, Mehraghdam et al. [22] model the VNFs placement problem as a Mixed Integer Quadratically Constrained Program with two different objectives: minimizing the number of activated nodes or the path latency. Kuo et al. [18] propose an algorithm for the VNFs placement and path selection problem with precedence constraints, while maximizing the number of accepted demands. Furthermore, Gupta et al. [14] provide a MIP formulation to model the placement of VNFs with chaining constraints. Their objective is to minimize network bandwidth consumption.

The closest version of the problem to ours, considering the problem with almost all its technical constraints is studied in Allybokus et al. [3]. The authors consider a generic version of the VNFs placement and routing problem for which many features are taken into account. They also proposed MIP formulations, either to minimize the total deployment cost or to minimize the number of rejected demands. A heuristic based on a continuous relaxation of one of the formulations is given, which appears to be very efficient on instances derived from the “Geant” network topology. However, the MIP models do not seem to be strong enough for direct resolution with the Cplex solver.

3. First extended formulation: the Path-based Formulation

In this section, we present the first extended MIP formulation, denoted by PF (which stands for “path-based formulation”) to model the VNFPRP. The formulation has been introduced in [25] and also used in [21] to generate heuristic solutions, by considering a compact model obtained from choosing a small but promising subset of columns. In this article instead we focus on developing an exact method for solving the path formulation, based on a Branch-and-Price procedure. In this model we use latency-constrained elementary path variables associated to each commodity to model routing decisions. In this section we discuss theoretical properties of this model, along with a derivation of a valid Lagrangian bound, whereas the details related to the B&P implementation are given in Section 6.

In the remainder of this paper we will assume that VNFs used by a commodity cannot be installed at the source node of the same commodity. This can be done without loss of generality, by introducing a dummy source node, say o_k, for each commodity $k \in C$, and connecting it to the source s_k with an arc (o_k, s_k) whose latency is set to 0. In the following we denote by $\Gamma^-(v)$ the set of all incoming neighbors of node v.

3.1. MIP formulation

The set of variables required in the path formulation is described in Table 2. Let us denote by $\mathcal{P}_k$ the set of all latency-constrained elementary paths associated with commodity $k \in C$. We assume that the set $\mathcal{P}_k$ is given and that, for each chosen path, the arcs composing it are known. Let t_{uv}^{pk} be the parameter that is equal to 1 if arc (u, v) belongs to path $p, p \in \mathcal{P}_k$, and equal to 0 otherwise.

The VNFPRP can be modeled as follows:

Variables		**Type**
λ_p^k	1, if path p associated with commodity k is chosen; 0, otherwise.	Binary
x_u^{fk}	1, if the virtual network function f is installed at or before node u for commodity k; 0, otherwise.	Binary
y_u^{fk}	1, if virtual network function f is installed at node u for commodity k; 0, otherwise.	Binary
w_u	1, if node u is activated; 0, otherwise.	Binary
z_u^f	number of VNF f installed at node u.	Integer

Table 2: Decision variables of the path formulation

$$(PF): \quad \min \quad \sum_{u\in N}\sum_{f\in F}\psi_u^f z_u^f + \sum_{u\in N}\psi_u w_u \tag{1}$$

$$\sum_{p\in\mathcal{P}_k}\lambda_p^k = 1 \qquad k\in C \qquad (\alpha_k) \tag{2}$$

$$\sum_{f\in F} z_u^f \leq c_u w_u \qquad u\in N \tag{3}$$

$$\sum_{k\in C} y_u^{fk} b_k \leq m_f z_u^f \qquad f\in F, u\in N \tag{4}$$

$$y_u^{fk} + y_u^{gk} \leq 1 \qquad k\in C, (f,g)\in\mathcal{A}^k, u\in N \tag{5}$$

$$\sum_{\substack{p\in\mathcal{P}_k\\(u,v)\in p}} t_{uv}^{pk}\lambda_p^k - 1 + x_v^{fk} - x_u^{fk} \leq y_v^{fk} \qquad k\in C, f\in F^k, (u,v)\in A \quad (\eta_{uv}^{fk}) \tag{6}$$

$$x_u^{gk} \leq x_u^{fk} \qquad k\in C, f,g\in F^k : f\prec_k g, u\in N \tag{7}$$

$$y_u^{fk} \leq x_u^{fk} \qquad k\in C, f\in F^k, u\in N \tag{8}$$

$$y_u^{fk} \leq \sum_{(v,u)\in A}\sum_{p\in\mathcal{P}_k} t_{vu}^{pk}\lambda_p^k \qquad k\in C, f\in F^k, u\in N \quad (\pi_u^{fk}) \tag{9}$$

$$\sum_{u\in N} y_u^{fk} \geq 1 \qquad k\in C, f\in F^k \tag{10}$$

$$x_u^{fk} = \begin{cases} 0, & u = s_k \\ 1, & u = d_k \end{cases} \qquad k\in C, f\in F^k \tag{11}$$

$$(\lambda, x, y, w) \text{ are binary} \tag{12}$$

$$z \text{ is integer} \tag{13}$$

Constraints (2) are the path constraints which ensure that exactly one elementary latency-constrained path $p \in \mathcal{P}_k$ is chosen for each commodity $k \in C$. Constraints (3) represent the node capacity constraints, which guarantee that the number of VNFs installed at each node $u \in N$ is bounded by its capacity c_u. Constraints (4) are the VNF capacity constraints they ensure that the volume of data treated by each function $f \in F$ should not exceed its capacity m_f. Constraints (5) are the conflict constraints and they guarantee that two VNFs in conflict are not installed at the same node $u \in N$. Constraints (6) are needed to link node installation variables (y), precedence variables (x) and path variables (λ): the left-hand-side is forced to 1 (implying that the function f is installed at the node v) if and only if (i) the path p passing through the arc (u, v) is chosen for the considered commodity k and (ii) the function f is installed at or before the node v and it is not installed at or before the node u. Constraints (7) impose the VNFs order for each commodity. Inequalities (8) link the precedence and the installation variables, x and y, and express the fact that if VNF f is installed at node u for the commodity k, then f is installed at or before the node u. Constraints (9) ensure that if a VNF $f \in F^k$ is installed at a node u for a given commodity k, then the associated routing path p must enter that node. Constraints (10) guarantee that all required functions for commodity $k \in C$ are installed at the graph nodes. Finally, constraints (11) guarantee that, for each commodity $k \in C$, no VNF is installed at or before the source node s_k and all VNFs are installed at or before the destination node d_k. Model (1)-(13) admits an exponential number of path variables; thus, a column generation (CG) procedure is needed to solve its continuous relaxation. In the following, we describe the pricing problem and discuss three possible procedures for its resolution.

3.2. The pricing problem

Let α, η and π be the dual variables associated with constraints (2), (6) and (9) respectively. Let DPF denote the dual formulation of the LP-relaxation of the model PF. The number of routing paths associated with each commodity can be exponential. Thus, the number of constraints associated to the path variables is exponential in the dual.

For each $k \in C$ and $p \in \mathcal{P}_k$, the dual constraint associated with the path variable λ_p^k is given as follows:

$$\alpha_k - \sum_{f \in F^k} [\sum_{(u,v) \in A} t_{uv}^{pk} \ \eta_{uv}^{fk} + \sum_{u \in N} \sum_{v \in \Gamma^-(u)} t_{vu}^{pk} \ \pi_u^{fk}] \quad \leq \quad 0, \qquad k \in C, p \in \mathcal{P}_k \tag{14}$$

$$\iff \qquad \alpha_k + \sum_{(u,v) \in A} \sum_{f \in F^k} (\pi_v^{fk} - \eta_{uv}^{fk}) t_{uv}^{pk} \quad \leq \quad 0, \qquad k \in C, p \in \mathcal{P}_k \tag{15}$$

As customary in column generation, the master problem is initialized with a subset of λ variables (resulting in the so-called *restricted master problem*), and then the additional variables necessary to solve the LP-relaxation of the model are generated on the fly by separating the associated dual constraints (15). The *pricing problem* then consists of finding for each commodity k, a path $p \in \mathcal{P}_k$ with negative reduced costs, i.e., a path p such that:

$$\alpha_k^* + \sum_{(u,v) \in p} \sum_{f \in F^k} (\pi_v^{*fk} - \eta_{uv}^{*fk}) \quad > \quad 0, \tag{16}$$

where $(\alpha^*, \eta^*, \pi^*)$ refers to a sub-vector of an optimal dual solution of the restricted master problem. This dual solution will be used for defining the pricing problems and computing the Lagrangian bound (cf. Section Appendix A.3).

Thus, for each commodity $k \in C$, a separate pricing problem is defined in order to find an $s_k - d_k$ latency-constrained elementary path of minimum cost. Cost per each arc is defined as

$$\tilde{c}_{uv} = \sum_{f \in F^k} (\eta_{uv}^{*fk} - \pi_v^{*fk}), \quad (u,v) \in A. \tag{17}$$

If we find a path $p \in P_k$ such that $\sum_{(u,v)\in p} \tilde{c}_{uv} - \alpha_k^* < 0$, the associated variable λ_p^k will be inserted in the restricted master problem. Based on the dual solution, the values defined by (17) may be negative. Therefore the pricing problem consists of finding an elementary shortest path satisfying latency constraints on a graph that may contain negative cycles. Hence, the pricing problem is *strongly* NP-hard [8]. Four different methods are proposed and computationally evaluated for solving this pricing problem (see Appendix).

3.3. Linear relaxation of path variables λ

Constraints $\lambda_p^k \in \{0,1\}$ are the integrality constraints guaranteeing that the latency-constrained path cannot be split. Together with constraints (2), they ensure that there is exactly one path used to route the flow for each commodity. In the following, we show that $\lambda_p^k \in \{0,1\}$ can be relaxed. Let PF' denote the model PF for which the integrality constraints associated with variables λ are replaced by: $\lambda_p^k \geq 0, \quad k \in C, p \in \mathcal{P}_k$.

Proposition 1. *If the relaxed path formulation PF' has an optimal solution with fractional λ values, then it must necessarily admit an integer solution with the same objective value.*

The proof of this result can be found in Appendix A.2.

Corollary 3.1. *Without loss of generality, constraints $\lambda_p^k \in \{0,1\}$, for all $k \in C, p \in \mathcal{P}_k$, can be replaced by $\lambda_p^k \geq 0$ in the model PF.*

Corollary 3.1 indicates that, when implementing a B&P procedure to solve the model PF, we can sidestep branching on the exponential variables λ and apply the regular branching scheme defined only for x, y, z, and w variables. In particular, this means that (apart from the change of the coefficients in the objective function), the pricing problem will not be affected by branching decisions.

4. Second extended formulation: the model DW

In this section, we present an alternative extended formulation for the VNFPRP, to which we refer as the model DW (where DW stands for Dantzig-Wolfe). First, we introduce the master problem and then, based on the master problem's dual solution, we describe the pricing problem and show how to calculate the value of the Lagrangian bound. At the end of this section, a branching scheme is proposed.

4.1. MIP formulation

In the following, we use the term *path-installation* to indicate a path p with pre-installed VNFs satisfying latency, conflict, and precedence constraints. The master problem aims to choose one path-installation per each commodity k while respecting node and function capacity constraints.

Variables		**Type**
τ_p^k	1, if path-installation p associated with commodity k is chosen; 0, otherwise.	Binary
w_u	1, if node u is activated; 0, otherwise.	Binary
z_u^f	number of VNF f installed at node u.	Integer

Table 3: Decision variables of the model DW

Let us denote by $\mathcal{T}_k$ the set of all path-installations associated with commodity k. To create our master problem, we need three families of variables, as described in Table 3.

The set $\mathcal{T}_k$ associated with each commodity k containing all feasible path-installations is supposed to be known. Thus, the placement of each VNF for each path-installation at network nodes is uniquely defined. Let us denote by a_u^{fpk} the parameter that is equal to 1 if the VNF f is used at node u for the path-installation p associated with commodity k; and that is equal to 0 otherwise.

The model DW is then given as:

$$(DW): \quad \min \quad \sum_{u\in N}\sum_{f\in F}\psi_u^f z_u^f + \sum_{u\in N}\psi_u w_u \tag{18}$$

$$\sum_{p\in\mathcal{T}_k}\tau_p^k = 1 \qquad k\in C \quad (\alpha_k) \tag{19}$$

$$\sum_{f\in F} z_u^f \leq c_u\, w_u \qquad u\in N \tag{20}$$

$$\sum_{k\in C}\sum_{p\in\mathcal{T}_k} a_u^{fpk}\tau_p^k b_k \leq m_f z_u^f \qquad f\in F,\quad u\in N \quad (\gamma_u^f) \tag{21}$$

$$\tau_p^k \in\{0,1\} \qquad k\in C,\quad p\in\mathcal{T}_k \tag{22}$$

$$w_u\in\{0,1\} \qquad u\in N \tag{23}$$

$$z_u^f\in\mathbb{N} \qquad u\in N,\quad f\in F \tag{24}$$

Constraints (19) represent the routing constraints ensuring that one path-installation is chosen for each commodity $k\in C$. Inequalities (20) represent the node capacity constraints. Constraints (21) are the VNF-capacity constraints.

Inequalities (18)-(24) constitute the master problem which admits an exponential number of variables. Therefore, a column generation procedure is needed to solve its continuous relaxation. The master problem is initialized with a subset of columns (called the restricted master problem), and the missing variables necessary to solve its linear relaxation are generated by separating the following dual constraints:

$$\alpha_k - \sum_{u\in N}\sum_{f\in F^k} a_u^{fpk} b_k \gamma_u^f \leq 0 \qquad k\in C,\quad p\in\mathcal{T}_k, \tag{25}$$

where we associate variables α and γ to constraints (19) and (21), respectively. The separation of constraints (25) represents the pricing problem. Let (α^*,γ^*) be components of the dual solution of the restricted master problem, the pricing problem consists of finding a commodity k and a path $p \in \mathcal{T}_k$ such that:

$$\alpha^*_k - \sum_{u \in N} \sum_{f \in F^k} a^{fpk}_u b_k \gamma^{*f}_u \quad > \quad 0.$$

4.2. The pricing problem

For each commodity k, we have one pricing problem that aims to find a path-installation. The left-hand-side in inequalities (25) characterizes the objective function of the pricing problem. We model the pricing problem as a MIP whose set of variables required is described in Table 4 (the index k is left out for simplicity).

Variables		**Type**
d^f_u	1, if virtual network function f is installed at or before node u; 0, otherwise.	Binary
h^f_u	1, if virtual network function f is installed at node u; 0, otherwise.	Binary
n_{uv}	1, if arc (u, v) belongs to the routing path; 0, otherwise.	Binary

Table 4: Decision variables of the pricing problem for the model DW

The MIP formulation of the pricing problem is given as follows:

$$\max \quad \alpha^*_k - \sum_{u \in N} \sum_{f \in F^k} h^f_u b_k \gamma^{*f}_u$$

$$\sum_{(u,v) \in A} n_{uv} - \sum_{(v,u) \in A} n_{vu} = \begin{cases} -1 & \text{if } u = d_k, \\ 1 & \text{if } u = s_k, \\ 0 & \text{otherwise.} \end{cases} \qquad u \in N \tag{26a}$$

$$\sum_{(u,v) \in A} n_{uv} l_{uv} \leq l_k \tag{26b}$$

$$h^f_u + h^g_u \leq 1 \qquad (f, g) \in \mathcal{A}^k, \quad u \in N \tag{26c}$$

$$n_{uv} - 1 + d^f_v - d^f_u \leq h^f_v \qquad f \in F^k, \quad (u, v) \in A \tag{26d}$$

$$d^g_u \leq d^f_u \qquad f, g \in F^k : f \prec_k g, u \in N \tag{26e}$$

$$h^f_u \leq d^f_u \qquad f \in F^k, \quad u \in N \tag{26f}$$

$$h^f_u \leq \sum_{(v,u) \in A} n_{vu} \qquad f \in F^k, \quad u \in N \tag{26g}$$

$$\sum_{u \in N} h^f_u \geq 1 \qquad f \in F^k \tag{26h}$$

$$d^f_{s_k} = 0 \qquad f \in F^k \tag{26i}$$

$$d^f_{d_k} = 1 \qquad f \in F^k \tag{26j}$$

$$(d, h, n) \text{ is binary} \tag{26k}$$

Constraints (26a) are the flow-preservation constraints ensuring that the path goes from the source node s_k to the destination node d_k. Inequalities (26b) represent the latency constraints. Inequalities (26c) are the anti-affinity constraints which ensure that two VNFs f and g in conflict are not installed at the same node u. Constraints (26d), (26e), (26i) and (26j) represent the precedence constraints. Constraints (26f) are the linking constraints between variables d and h, the right-hand-site is forced to 1 in order to ensure that if VNF f is used at node u, then it is used at or before u. Inequalities (26g) link variables h and n, they guarantee that the routing path enter all nodes at which VNFs are installed. Finally, constraints (26h) guarantee that all required VNFs for the current commodity are installed at nodes.

Proposition 2. *The binary constraints imposed on the arc variables n in the pricing problem* (26a)-(26k) *can be relaxed and replaced by $n_{uv} \geq 0$, for all $(u, v) \in A$.*

Proof . Observe that variables n do not appear in the objective function, and that the location variables h (which basically determine the value of the solution) remain binary. Therefore, if there exists a feasible solution of the pricing problem with fractional n values, there also exists a latency constrained path (corresponding to binary n values) which satisfies all the constraints (26a)-(26k). To show the latter result, one has to follow similar arguments as those given in the proof of Proposition 1 provided in Appendix A. □

4.3. Branching on τ variables

The LP-relaxation of the Dantzig-Wolfe formulation solved by CG procedure is not necessarily integral. Furthermore, applying the Branch-and-Bound algorithm on the restricted master problem with only the generated columns at the root node will not guarantee a feasible solution and so an optimal solution. Moreover, at each branching node, there may exist new columns with a negative reduced cost which should be added to the master problem. Therefore, in order to find an optimal integer solution, we should generate columns at each branching node.

Various branching schemes, specific (like the one proposed below) or generic (see e.g., [32]), can be used to generate integer solutions using column generation procedure embedded within the Branch-and-Bound algorithm. The resulting algorithm is called *Branch-and-Price*.

In the following, a commodity $k \in C$ is called *fractional* if it admits a fractional τ variable. For a path-installation $p \in \mathcal{T}_k$, we use the notation $u \in p$ to indicate that the path-installation p passes through the node u. Recall that, the notation $a_u^{fpk} = 1$ is used to indicate that the path-installation p passes through the node u on which the VNF f is installed for commodity k.

Proposition 3. *For any given LP-solution of the (restricted) master problem with fractional τ variables, at least one of the following cases is valid for each fractional commodity $k \in C$:*

Case 1. *There exist two nodes $u, v \in N \setminus \{s_k\}$ satisfying:* $0 < \sum_{\substack{p \in \mathcal{T}_k \\ u \in p, v \in p}} \tau_p^k < 1$

Case 2. *There exist a function $f \in F^k$ and a node $u \in N \setminus \{s_k\}$ satisfying:* $0 < \sum_{\substack{p \in \mathcal{T}_k \\ a_u^{fpk} = 1}} \tau_p^k < 1$

Proof . Let us suppose that k is a fractional commodity but neither **Case 1** nor **Case 2** holds. Hence, we have:

$$\sum_{\substack{p \in \mathcal{T}_k \\ u \in p, v \in p}} \tau_p^k \in \{0, 1\} \qquad u, v \in N \tag{27}$$

$$\sum_{\substack{p \in \mathcal{T}_k \\ a_u^{fpk}=1}} \tau_p^k \in \{0, 1\} \qquad u \in N, f \in F^k. \tag{28}$$

From (28) we can distinguish two cases: (a) there exists a node $u \in N$ and a function $f \in F^k$ such that $\sum_{\substack{p \in \mathcal{T}_k \\ a_u^{fpk}=1}} \tau_p^k = 1$, or (b) $\sum_{\substack{p \in \mathcal{T}_k \\ a_u^{fpk}=1}} \tau_p^k = 0$, for each $u \in N$ and $f \in F^k$.

(a) By constraints (19), and because k is a fractional commodity, all path-installations of $\mathcal{T}_k$ in the solution are fractional. Let $p_1 \in \mathcal{T}_k$ be a fractional path-installation passing through node u on which VNF f is installed (i.e., $0 < \tau_{p_1}^k < 1$). As $\sum_{\substack{p \in \mathcal{T}_k \\ a_u^{fpk}=1}} \tau_p^k = 1$, there must exist another fractional path-installation $p_2 \in \mathcal{T}_k \setminus \{p_1\}$ passing through node u on which VNF f is installed. Since $p_1 \neq p_2$, we have two cases:

(I) p_1 and p_2 pass through the same nodes but with different function installations, i.e., there exists at least one VNF $g \in F^k \setminus \{f\}$ installed on a different node. Let denote by v (resp. w, $v \neq w$) the node belonging to the path-installation p_1 (resp. p_2) on which g is installed. As the hypothesis (28) is valid for any VNF in F^k and any node in $N \setminus \{s_k\}$, $\sum_{\substack{p \in \mathcal{T}_k \\ a_v^{gpk}=1}} \tau_p^k \in \{0, 1\}$ must hold. Given that $\tau_{p_1}^k > 0$, then (1) $\sum_{\substack{p \in \mathcal{T}_k \\ a_v^{gpk}=1}} \tau_p^k > 0$. Thus $\sum_{\substack{p \in \mathcal{T}_k \\ a_v^{gpk}=1}} \tau_p^k = 0$ cannot hold. Accordingly, $\sum_{\substack{p \in \mathcal{T}_k \\ a_v^{gpk}=1}} \tau_p^k = 1$. We know that $0 < \tau_{p_2}^k < 1$ and that g is not installed on v for p_2, we will have: (2) $\sum_{\substack{p \in \mathcal{T}_k \\ a_v^{gpk}=1}} \tau_p^k < 1$. Therefore, (1) and (2) contradict hypothesis (28).

(II) p_1 and p_2 pass through at least one different node, i.e., there should exist another node $v \neq u$ belonging to p_2 and not to p_1. We notice that VNF f is installed only on node u for both p_1 and p_2, and that another VNF $g \neq f$ can or not be installed on node v belonging to the path-installation p_2. Since p_2 contains u and v and $\tau_{p_2}^k > 0$, this implies: (i) $\sum_{\substack{p \in \mathcal{T}_k \\ u \in p, v \in p}} \tau_p^k > 0$. Moreover, as $\sum_{\substack{p \in \mathcal{T}_k \\ a_u^{fpk}=1}} \tau_p^k = 1$ and $\tau_{p_1}^k > 0$ and we know that path-installation p_1 does not pass through node v, then the value $\tau_{p_1}^k$ can be deleted from the following sum: $\sum_{\substack{p \in \mathcal{T}_k \\ u \in p, v \in p}} \tau_p^k$, which implies that (ii) $\sum_{\substack{p \in \mathcal{T}_k \\ u \in p, v \in p}} \tau_p^k < 1$. Therefore, (i) and (ii) contradict hypothesis (27).

(b) Recall that for each commodity k, we assume that $F^k \neq \emptyset$ (otherwise the commodity can be pre-processed and eliminated). This, together with constraints (19), contradicts the assumption that for all functions $f \in F^k$, no path-installation $p \in \mathcal{T}_k$ is chosen such that $a_u^{fpk} = 1$ (i.e., it contradicts the hypothesis $\sum\limits_{\substack{p \in \mathcal{T}_k \\ a_u^{fpk}=1}} \tau_p^k = 0$).

Therefore, the result holds. ■

5. Strengthening inequalities

In this section, we derive several families of valid inequalities that can strengthen the LP-bounds of both proposed extended formulations. We first present inequalities that can be used to directly enhance the model PF. We then present inequalities that are valid for both models and that can be exploited in case a function's capacity is smaller than the respective traffic demand. We close this section by explaining how some of inequalities proposed for the model PF could be used within the Dantzig-Wolfe decomposition to strengthen the model DW.

5.1. Valid inequalities for the model PF

Proposition 4. *Inequalities (29) and (30) are valid for the VNFPRP:*

$$y_u^{fk} \leq w_u, \quad u \in N, \quad k \in C, \quad f \in F^k. \tag{29}$$

$$y_u^{fk} + y_u^{gk} \leq w_u, \quad u \in N, \quad k \in C, \quad (f, g) \in \mathcal{A}^k. \tag{30}$$

Proof . For a given commodity $k \in C$, if the VNF $f \in F^k$ is installed at node u, then node u should be activated. If two VNFs f and g are in conflict for a commodity $k \in C$, at most one of them can be installed at the same node u, if u is activated. ■

Let $D_k = (F^k, E)$ be the conflict graph associated with commodity k, $k \in C$, where nodes in D_k represent the VNFs $f \in F^k$. An edge $e \in E$ between two nodes f and g in D_k represents the fact that f and g are in conflict. Let $\mathcal{D}_k$ denote the set of all maximal cliques in D_k and let $\omega(D_k)$ be the *clique-number* (i.e., the size of the maximum clique) in the conflict graph. Linear inequalities (30) can be generalized for each clique Q in $\mathcal{D}_k$.

Proposition 5. *Inequalities (31) are valid for the VNFPRP:*

$$\sum_{f \in Q} y_u^{fk} \leq w_u, \quad u \in N, \quad k \in C, \quad Q \in \mathcal{D}_k. \tag{31}$$

Proof . For a given commodity $k \in C$, nodes in Q represent the set of VNFs in conflict, i.e., they cannot be installed at the same node. Therefore, at most one VNF in Q can be installed at node u, provided that the node u is activated. ■

Given a commodity $k \in C$ and a node $u \in N$, if there is a unique path p going from s_k to u in G, let A_p be the arcs belonging to the path p.

Proposition 6. *For a given commodity $k \in C$, and a node $u \in N$, if there exists a unique path from s_k to u in G, then inequalities (32) are valid for the VNFPRP:*

$$\sum_{f \in Q} x_u^{fk} \leq |A_p|, \quad u \in N, \quad k \in C, \quad Q \in \mathcal{D}_k \ : \ |A_p| < |Q|. \tag{32}$$

Proof . Given a fixed commodity k, let u be a node for which we have a unique path p going from s_k to u, and let Q be a clique in the graph D_k, such that $|A_p| < Q$. The number of VNFs in conflict installed at or before node u should be less than or equal the number of arcs in the path p; otherwise, two or more functions in conflict need to be installed at the same node, which leads to an infeasible solution. ■

Proposition 7. *Inequalities (33) are valid for the VNFPRP.*

$$\sum_{u \in N} w_u \geq \max\{1, \max_{k \in C} \omega(D_k)\}. \tag{33}$$

Proof . Recall that each commodity requires at least one VNF. As all required VNFs should be installed at graph nodes, at least one node in the graph must be activated. Furthermore, if there is a conflict between VNFs associated with one commodity k, then the number of activated nodes should be at least equal to the maximum number of VNFs in conflict, which is the clique number of D_k. ■

Proposition 8. *Inequalities (34) are valid for the VNFPRP.*

$$\sum_{u \in N} \sum_{f \in Q} y_u^{fk} \geq |Q|, \quad k \in C, \quad Q \in \mathcal{D}_k. \tag{34}$$

Proof . From inequalities (10), all VNFs required for each commodity $k \in C$ should be installed at graph nodes, thus the number of nodes necessary to install VNFs in conflict should be at least equal to the number of VNFs in conflict, which is equal to the size of the cliques from $\mathcal{D}_k, k \in C$. ■

Let $C^u \subseteq C$ be a subset of commodities for which there exists at least one latency-constrained path visiting node u, (i.e., if $k \notin C^u$, this means that all paths associated with k do not enter the node u). Let N^k be the set of nodes belonging to at least one latency-constrained path associated with commodity $k \in C$ (this can be checked in polynomial time using a min-cost flow algorithm for example).

Proposition 9. *The following inequalities are valid for the VNFPRP.*

$$\sum_{k \in C \setminus C^u} \sum_{f \in F^k} y_u^{fk} = 0, \quad u \in N, \tag{35}$$

$$\sum_{k \in C \setminus C^u} \sum_{f \in F^k} x_u^{fk} = 0, \quad u \in N. \tag{36}$$

Moreover, inequalities (37) are valid and dominate inequalities (10);

$$\sum_{u \in N^k} y_u^{fk} \geq 1, \quad k \in C, \quad f \in F^k, \tag{37}$$

$$\sum_{f \in F} z_u^f = 0, \quad u \in N \setminus \{\cup_{k \in C} N^k\}, \tag{38}$$

$$\sum_{u \in N \setminus \{\cup_{k \in C} N^k\}} w_u = 0. \tag{39}$$

Proof . If there exists a node $u \in N$ which is not visited by any commodity $k \in C$, then no function $f \in F^k$ can be installed at u (35), thus at or before u (36). Therefore, VNFs associated with commodity $k \in C$ can be installed only at nodes in N^k (37). In consequence, u cannot be activated (39), so the number of VNFs installed on it is equal to zero (38). ■

Proposition 10. *Inequalities (40) are valid for the VNFPRP:*

$$z_u^f \geq \lceil \frac{b_k}{m_f} \rceil y_u^{fk}, \quad k \in C, \quad f \in F^k, \quad u \in N. \tag{40}$$

Proof . The number of VNFs installed at node u is at least equal to the number of VNFs needed to handle one commodity $k \in C$. ■

Proposition 11. *Inequalities (41) are valid for the VNFPRP:*

$$x_u^{fk} \leq 1 - y_{d_k}^{fk}, \qquad k \in C, \quad u \in \Gamma^-(d_k), \quad f \in F^k, \tag{41}$$

Proof . If a VNF f associated with commodity $k \in C$ is installed at the destination node, then, this function cannot be installed at or before any predecessor of d_k. ■

5.2. Strengthening inequalities for both models

Besides inequalities (33), (38) and (39) which are also valid for the model DW, in the following we propose additional inequalities that involve only z and w variables, and are therefore valid for both formulations studied in this paper. With valid inequalities given in Proposition 10 we address the setting in which the capacity of a function is not sufficient to handle the full demand of a given commodity (i.e., multiple copies of the same function need to be installed). Proposition 12 provides further generalizations of this setting.

Proposition 12. *Node capacity constraints (3) do not define facets of the polyhedron of the VNF-PRP if there exists a node u, such that $c_u > \sum\limits_{k \in C} \sum\limits_{f \in F^k} \lceil \frac{b_k}{m_f} \rceil$.*

1. *Therefore, inequalities (42) are valid for the VNFPRP and dominate inequalities (3).*

$$\sum_{f \in F} z_u^f \leq \sum_{k \in C} \sum_{f \in F^k} \lceil \frac{b_k}{m_f} \rceil w_u, \quad u \in N. \tag{42}$$

2. *Moreover, if $|C^u| < |C|$, then the linear inequalities (43) dominate (42):*

$$\sum_{f \in F} z_u^f \leq \sum_{k \in C^u} \sum_{f \in F^k} \lceil \frac{b_k}{m_f} \rceil w_u, \quad u \in N. \tag{43}$$

3. *In addition if there exists a conflict between functions in F^k for a given commodity $k \in C$, with $m_{f_1} \leq m_{f_2} \leq \cdots \leq m_{f_{|Q|}}$, $Q \in \mathcal{D}_k$ and $c_u \geq \sum\limits_{k \in C} \sum\limits_{f \in F^k} \lceil \frac{b_k}{m_f} \rceil$, then inequalities (43) are dominated by the following inequalities.*

$$\sum_{f \in F} z_u^f \leq \sum_{k \in C} \sum_{Q \in \mathcal{D}_k} [(\sum_{f \in F^k} \lceil \frac{b_k}{m_f} \rceil) - \sum_{\substack{i=2 \\ f_i \in Q}}^{|Q|} \lceil \frac{b_k}{m_{f_i}} \rceil] w_u, \quad u \in N. \tag{44}$$

Proof .

1. If there exists a node u having enough capacity to install VNFs required for all commodities, then the number of VNFs needed to treat all commodities bandwidth is bounded by $\sum_{k\in C}\sum_{f\in F^k}\lceil\frac{b_k}{m_f}\rceil$, which is the maximum number of VNFs necessary to handle all demands.
2. Only VNFs associated with commodities having at least one path passing through a node u can be installed at node u.
3. The number of VNFs installed at node u when the conflict constraints are considered is bounded by the maximum number of copies needed to install the VNFs with the smallest capacity for each commodity (i.e., in the worst case we will keep the VNFs with the maximum instantiation installed at node u and install other VNFs at the other nodes). ■

5.3. *Strengthening the model DW*

Valid inequalities proposed for the model PF can be "translated" into valid inequalities for the model DW. In this subsection we illustrate how this can be done for inequalities (29), (30) and (40). The remaining inequalities can be translated accordingly. In order to add (29), (30) and (40) to the master problem of the model DW, we rewrite them using parameters a as (45), (46) and (47), respectively:

$$\sum_{p\in\mathcal{T}_k} a_u^{fpk}\tau_p^k \le w_u, \qquad k\in C, \quad u\in N, \quad f\in F^k \tag{45}$$

$$\sum_{p\in\mathcal{T}_k} (a_u^{fpk}+a_u^{gpk})\tau_p^k \le w_u, \qquad k\in C, \quad u\in N, \quad (f,g)\in\mathcal{A}^k \tag{46}$$

$$z_u^f \ge \lceil\frac{b_k}{m_f}\rceil \sum_{p\in\mathcal{T}_k} a_u^{fpk}\tau_p^k, \quad k\in C, \qquad f\in F^k, \quad u\in N \tag{47}$$

Adding these valid inequalities generates new (non-negative) dual variables in the dual of the master program, that we denote by δ, η and φ, respectively. Thus, dual constraints (25) need to be replaced by the following inequalities:

$$\alpha_k - \sum_{u\in N}\left[\sum_{f\in F^k}(b_k\gamma_u^f + \lceil\frac{b_k}{m_f}\rceil\varphi_u^{fk} + \delta_u^{fk})a_u^{fpk} + \sum_{(f,g)\in\mathcal{A}^k}\eta_u^{fgk}(a_u^{fpk}+a_u^{gpk})\right] \le 0, \quad k\in C, p\in\mathcal{P}_k$$

6. Two Branch-and-Price algorithms

In this section, we explain major implementation ingredients of the two Branch-and-Price algorithms derived from the models PF and DW, respectively. Further implementation details regarding specific pricing algorithms and derivations of Lagrangian bounds are given in the Appendix.

6.1. *Generic column generation framework*

Initialization. The restricted master problem of both models is initialized by a subset of columns building a heuristic solution which is obtained in the initialization phase of the algorithm (see Section 6.3). If no solution has been found during a time-limit, the CG framework is initialized with an artificial column whose cost is set to a very large number.

Bounding. At each iteration of the column generation procedure, the restricted master problem is solved, and a dual solution is provided. Accordingly, the objective function of the pricing problem for each commodity k is updated, and the pricing problem is solved. Depending on the pricing strategy (see Appendix A.5) multiple columns per commodity having negative reduced costs (or at most one) are added to the restricted master problem. During this process, to reduce the number of CG iterations, we keep track of the Lagrangian bound (see Appendix A.3 and Appendix A.4) and compare it to the objective value of the current restricted master problem. If the difference between the two values is smaller than ε, the column generation procedure is stopped and we resort to branching.

Pricing for the model PF. We consider three methods for solving the pricing problem for the formulation PF: 1) We utilize dynamic programming; 2) Based on Yen's algorithm [33], we derive a reduced cost method, and use it in two different ways; and 3) We model the pricing problem as a MIP that we solve using an off-the-shelf solver. These strategies are explained and computationally evaluated in the Appendix.

Pricing for the model DW. In order to price the columns associated with the DW formulation, the MIP model presented in Section 4.2, is solved for each commodity $k \in C$, using an off-the-shelf solver. Thus, at each iteration of the column generation procedure at most one column per commodity with negative reduced cost is generated by the pricing problems. As in the case of the model PF, the column generation phase terminates when no more columns with a negative reduced cost are found, or when the gap between the current value of the restricted master problem and the Lagrangian bound is smaller than ε.

6.2. Branching

At the end of the column generation phase, the integrality of the solution of the relaxed master problem is verified. If the current solution is not integer, we branch on the most fractional variable, applying the BFS (Breadth-First Search) based branching strategy. Specifically, we explore all the nodes of the same level in the branching tree before moving to the next level. In our implementation the algorithm explores all nodes admitting a fractional feasible solution of the same level by applying the respective branching scheme described below. For each branching node with a fractional solution, two children nodes are created and saved in a queue. The nodes in the queue are explored using the FIFO (First In First Out) method. A global lower bound is calculated at each level. In our preliminary experiments, we also tried the diving strategy as an alternative to the BFS-based branching. Whereas diving is very useful when searching for feasible solutions (see e.g., [10, 12, 17]), in our case this strategy did not prove useful, because a high-quality feasible solution is used to initialize the CG procedure (see Section 6.3).

Branching scheme for the model PF. In Corollary 3.1 we showed that the binary constraints on λ variables in the PF model can be relaxed to $\lambda \geq 0$. Hence, in our BP implementation of the PF model, we branch only on the (x, y, z, w) variables.

Different branching schemes have been tested for the model PF; The one outperforming all others is to branch first on the most fractional w variables (by imposing either $w \geq 1$ or $w \leq 0$), secondly on z by setting either $z \geq \lceil z^* \rceil$ or $z \leq \lfloor z^* \rfloor$, thirdly on y which are forced to be $y \geq 1$ or $y \leq 0$ and finally on x variables using $x \geq 1$ or $x \leq 0$.

Branching scheme for the model DW. In our branching scheme for the model DW we start branching on z variables by setting $z \geq \lceil z \rceil$, or $z \leq \lfloor z \rfloor$ and then on w variables by setting $w \leq 0$, or $w \geq 1$. When z and w variables are integers, we continue by branching on the τ variables. Given that the path-installation variables are generated as and when they are needed, we follow the specific branching scheme for τ variables proposed in Proposition 3. Specifically, if we find a pair of two distinct nodes $u, v \in N$ such that $\sum\limits_{p \in \mathcal{T}_k : u \in p, v \in p} \tau_p^k$ is fractional we create two branches by imposing constraints that limit the respective sum to 0, or 1, respectively. If none such pair can be found, we search for a node $u \in N$ and a function $\bar{f} \in F^k$ such that $\sum\limits_{p \in \mathcal{T}_k : a_u^{\bar{f}pk} = 1} \tau_p^k$ is fractional, and create two branches correspondingly.

A branching scheme is said to be *complete*, if it can generate any feasible solution. From Proposition 3 we conclude that our branching scheme proposed for the model DW is complete, as at least one of the two cases should hold for any fractional commodity.

6.3. Heuristics

Before entering the B&P phase, we generate a heuristic solution which provides a high-quality upper bound and a promising set of columns that we use to initialize the CG procedure. For the model PF, we employ the MIP-based heuristic presented in [25]. The heuristic solves a compact model derived from the formulation PF in which only a small subset of latency-constrained paths is considered. To obtain this subset, we run Yen's algorithm [33] which provides κ elementary paths between two nodes, sorted from the shortest to the longest one. The number of generated paths per commodity is capped by κ, and we let $\kappa \in \{10, 15, 20, \ldots, 50\}$. As soon as the underlying MIP-based heuristic finds a feasible solution for a fixed κ value, we stop. However, for some instances, even for $\kappa = 50$ we fail to find a feasible solution.
For the model DW, we start with an artificial column, price-in the columns with negative reduced cost, and then convert the obtained linear program into a MIP.
In both cases, there is a time limit after which this MIP-based heuristic initialization is aborted.

7. Computational results

In this section we analyze the scalability and the efficiency of the two proposed B&P algorithms and show the benefits of the valid inequalities defined in Section 5. The B&P algorithms are compared to two other exact methods using the commercial solver Cplex: the first one is a compact MIP formulation (denoted by `C` and provided in the Appendix) proposed in [21] and the second one is the Automatic Benders approach by Cplex [4] applied to the model `C` in which a family of flow variables is linearly relaxed. Our experiments are designed so as to evaluate the effectiveness and the performance of the proposed extended formulations in terms of CPU time, quality of bounds and final gaps. Eventually we also measure the advantage of the proposed valid inequalities in improving the LP-bounds and reducing the final gaps.

All the experiments described in this section were made using a computer with Intel(R)Xeon(R) CPU E5-2650 v2 processor clocked at 2.60GHz, 32 cores, 2 threads per core and 252GB RAM, under Linux operating system. All methods are implemented using the Python API for Cplex, which is run in single-thread mode with a default memory limited to 20GB. All Cplex parameters were set to their default values. A default time limit of one hour is set for each tested instance. For the initialization heuristic used within the B&P algorithm (cf. Section 6.3), the time limit is fixed to 900 seconds.

The following settings represent all tested methods in our computational experiments:

- `PF`: The Branch-and-Price implementation of the model PF with the best-performing pricing method (more details are given in Appendix A.5);
- `PF+VI`: The setting `PF` in which Valid Inequalities (29)– (41) and (42)–(44) presented in Section 5 are additionally used to initialize the model;
- `DW`: The Branch-and-Price implementation of the model DW presented in Section 4;
- `DW+VI`: The setting `DW` in which Valid Inequalities (33), (38), (39), (42)– (47) from Section 5 are added to the model;
- `C`: compact formulation based on the original model proposed in [21] with a modified objective function and extended with node-capacity and VNF-capacity constraints (see Appendix);
- `AB`: Branch-and-Benders-cut approach derived from the formulation `C` (in which binary flow variables are linearly relaxed and projected out). The implementation is based on automatic Benders decomposition of Cplex [4].

7.1. Benchmark instances

In order to perform our experiments, we have generated a set of instances derived from the SNDlib library [29] of telecommunication networks. An instance from SNDlib contains a graph defined by its set of nodes, its set of arcs and a set of commodities. For each commodity, a source node, a destination node and a bandwidth value are given. Based on a longitude and latitude values which are provided for every node, and using the fibre propagation delay per km (see, e.g., [7, 19]), we calculate the latency value of each arc in this graph (for more details, see [25]).

The remaining parameters required for our settings are generated as follows (see also [25]). To construct the set of VNFs, we consider six Virtual Network Functions typically employed in service function chaining [16, 28]: F = {*NAT: Network Address Translator, FW: Firewall, TM: Traffic Monitor, WOC: WAN Optimization Controller, IDPS: Intrusion Detection Prevention System, VOC: Video Optimization Controller*}. For each generated instance, we divide the set of commodities in five categories defining the offered services: *Online Gaming*, *Video Streaming*, *Voice over IP*, *Web Services*, and *Other Services*. Accordingly, a latency value l_k and a set of chained service functions F^k are defined. Table 5 shows the considered services and the associated latency value and SFC (representing a partial order between VNFs). For each service we consider at most one anti-affinity constraint (AAC) between VNFs. We suppose that we cannot install Firewall and Network Address Translator at the same node. For each commodity $k \in C$, first the length of the shortest path $l(SP_k)$ between source and destination is calculated (with respect to arc latencies). Accordingly, the demand k is randomly assigned to one of the categories, for which the latency value (cf. the first column in Table 5) is bigger than $l(SP_k)$. Namely, if $l(SP_k) = 55ms$, this implies that the commodity can be assigned to any of the five categories, and we randomly choose one. Each input graph from the SNDlib corresponds to one *instance-type* from SNDlib: {"Abilene", "Atlanta", "Di-yuan","France", "Geant", "Newyork", "Nobel-eu", "Nobel-germany", "Nobel-us", "Pdh", "Polska", "Ta1"}, and for each instance-type we generate ten different instances by randomly assigning demands to services. The generated instances can be found in the `github` repository [24].

Latency value	Service	SFC
$\leq 60\ ms$	Online Gaming (O-G)	NAT-FW-TM-WOC-IDPS
$\leq 100\ ms$	Video Streaming (V-S)	NAT-FW-TM-VOC-IDPS
	VoIP	NAT-FW-TM-FW-NAT
$\leq 500\ ms$	Web Services (W-S)	NAT-FW-TM-WOC-IDPS
$\leq \infty\ ms$	Other services (O-S)	NAT-FW-TM-WOC-VOC

Table 5: Five services and their respective SFCs and latency values.

7.2. Obtained results

In the sequel we summarize the major results obtained by our computational study. Tables with more detailed information provided per each instance can be found in Appendix A. This subsection is divided into two parts. The first part compares the LP-relaxation bounds generated by applying the column generation algorithm to the models PF and DW, respectively. The comparison between the four different pricing methods proposed for the model PF in terms of CPU time, number of added columns and generated iterations can be found in Appendix A.5.1. After determining the best configuration (i.e., the best pricing method) for the model PF, we focus on the improvement of the LP-gap, with respect to the LP-relaxation bounds provided by the compact formulation (`C`). We compare the two extended formulations, with and without adding valid inequalities.

In the second part of the empirical study, we compare the two proposed B&P algorithms (`PF` and `DW`) with two other alternative MIP approaches, namely the Compact formulation (`C`) and the Branch-and-Benders-cut (`AB`). We report the overall CPU time in seconds, and compare the quality of lower bounds after reaching the time limit. We also report the number of added columns and generated iterations during the B&P algorithms.

We recall that the time limit is equal to 1 hour for all instances solved with exact methods, which does not include the time for the initialization heuristic (fixed to 900 seconds). We notice that the heuristics can find a feasible solution for all solved 120 instances.

7.2.1. Comparison of LP-relaxation bounds

Based on the comparison of CPU times between the four pricing methods proposed for the Path formulation (cf. Appendix A.5.1), in the following, we will consider `Red Cost 2` as our *default pricing method* for the model PF. This setting will be denoted by `PF` in the remainder of this section.

We now turn our attention to the comparison of the quality of LP-relaxation bounds and the CPU time required for solving LP-relaxation of the formulations PF, DW and `C`. We also show the relative improvement of lower bounds, with respect to the LP-bounds provided by `C`, by both extended formulations with and without adding valid inequalities.

Figure 1 provides a cumulative chart in which we report the CPU time for all 120 instances from our test bed. We observe that in less than 31 seconds the LP-solution for any of the considered 120 instances can be found by the compact formulation `C`. The CPU time consumed by the model PF is below 1000 seconds. Finally, the model DW can solve only 91 instances at the root node without exceeding the time limit.

Figure 2 depicts the CPU time consumed by `PF`, `PF+VI`, `DW` and `DW+VI` settings at the root node of the branching tree. We observe that adding valid inequalities to both formulations increases the CPU time. Valid inequalities slow down the resolution at the root node for the model PF;

75% of instances were solved with CPU time below 300secs, whereas after adding valid inequalities, the same percentage of instances needs up to 1025secs. Moreover, the overall number of solved instances decreases from 120 to 109. Similarly, for the model DW, after adding valid inequalities the number of instances for which LP-relaxation can be solved drops from 92 to 66 instances.

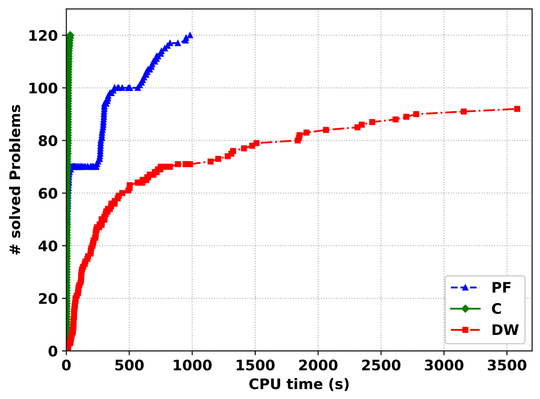


Figure 1: CPU time needed to solve the LP-relaxation of the `PF`, `C` and `DW` formulation.

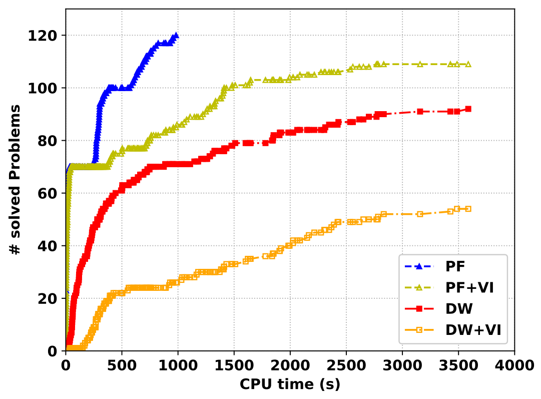


Figure 2: CPU time needed to solve LP-relaxation for `PF` and `DW` with and without valid inequalities.

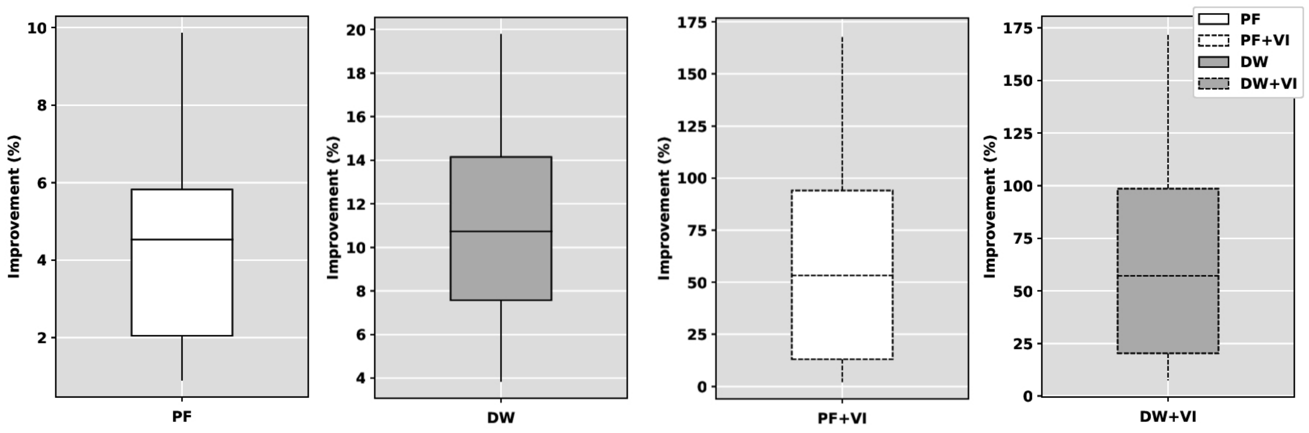


Figure 3: Relative improvement of LP-relaxation bounds of `C` by using `PF` and `DW` with/without valid inequalities.

In the following, we focus on 52 instances whose LP-relaxation could be solved at the root node by `PF, PF+VI, DW, DW+VI` without exceeding the time limit (2 among 54 instances solved by `DW+VI` are not solved by `PF+VI`). [1] We compare the relative improvement of lower bounds with respect to the LP-bounds obtained by the compact formulation `C`. Figure 3 illustrates the relative improvement of lower bounds, calculated as ((LB$_{\mathtt{a}}$ - LB$_{\mathtt{C}}$) / LB$_{\mathtt{C}}$) * 100, with $a \in \{$`PF, PF+VI, DW, DW+VI`$\}$. We

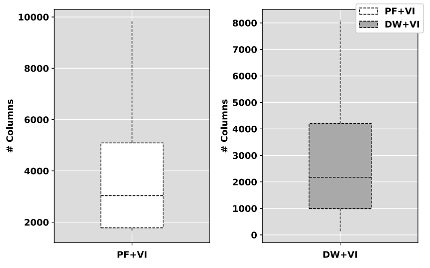
[1]By looking into the structure of the remaining 68 instances, we realize that it is a combination of graph density (i.e., the ratio $|A|/|N|$) and the number of commodities, that makes some instance types more challenging than the others. These 68 instances have either the ratio $|A|/|N| > 4$, or the large number of commodities with respect to the number of nodes ($|C|/|N| > 10$). Given that graph density and the number of commodities affect both, the number of inequalities in the master and in the pricing problem, this explains why such instances become computationally more difficult to solve.

observe that the lower bound at the root node could be improved by between 1% to 14% (resp. by between 3% to 29%) with `PF` (resp. `DW`) method for all considered instances without adding valid inequalities. Moreover, the effects of adding the valid inequalities to both extended formulations are shown. We notice that valid inequalities significantly improve the quality of LP-relaxations bounds at the root node. The LP-relaxation bounds of the `C` formulation are improved from 6% to 90% for 75% of instances solved by the setting `PF+VI`. This improvement ranges between 14% and 100% for 75% of instances solved by `DW+VI`.

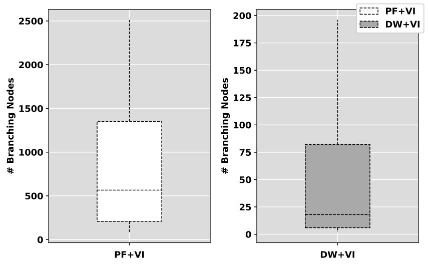
7.2.2. *Comparison between the proposed B&P algorithms, a compact formulation and its Branch-and-Benders-cut implementation*

In this section, we first look at some performance indicators of our two Branch-and-Price algorithms (settings `PF+VI` and `DW+VI`, respectively), and then we compare them against two alternative exact approaches (the first one is based on solving the compact formulation, and the second one is based on solving its Benders reformulation).

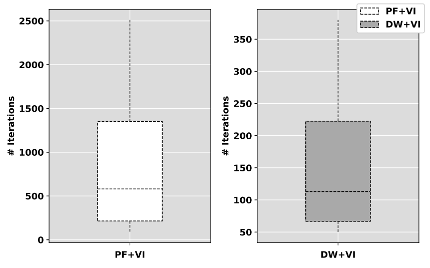
Box-plots given in Figures 4(a), 4(b) and 4(c) compare the overall number of added columns during the B&P algorithm, the number of branching nodes and the number of generated iterations for `PF+VI` and `DW+VI`, respectively. We observe that the setting `PF+VI` needs more columns and more iterations and also branches more than `DW+VI` (within the same time limit). This is not surprising, due to the following observations: 1) The vast number of added columns for `PF+VI` is explained by the fact that we are adding all columns with negative reduced cost at each iteration of the CG procedure. This is in contrast to `DW+VI` where we are using MIP-based pricing method to add only the most violated ones; 2) The number of variables that are required to be integer in the model PF is much larger compared to the model DW, which also explains the larger number of branching nodes for the setting `PF+VI`; 3) Finally, the CPU time required for a single pricing iteration is much lower for the setting `PF+VI` than for `DW+VI`.



(a) # Columns



(b) # Nodes



(c) # Iterations

Figure 4: The number of added columns, generated nodes and CG iterations for the settings `PF+VI` and `DW+VI`.

Finally, we compare the two proposed B&P algorithms (settings `PF+VI` and `DW+VI`) against the following two alternatives:

- Compact formulation (denoted as setting `C`) based on the original model proposed in [21] with a modified objective function and capacity constraints (see Appendix).
- Branch-and-Benders-cut approach derived from the formulation `C` and implemented using automatic Benders decomposition of Cplex (setting `AB`).

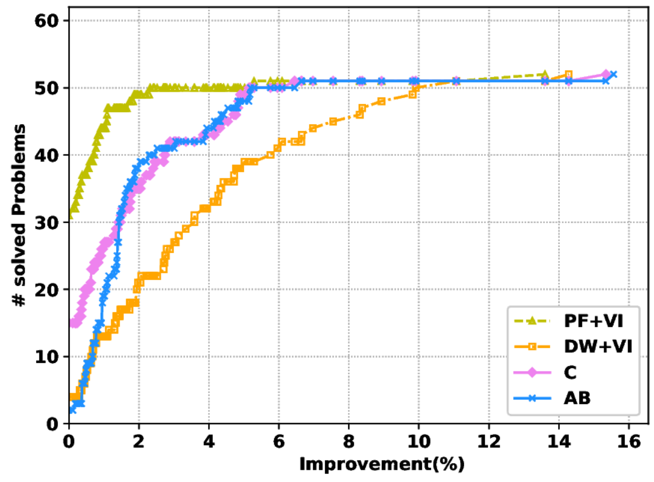


Figure 5: Relative improvement of the global lower bound with respect to $GLB_{\min}$.

As we are comparing four exact methods, we focus on the quality of global lower bounds obtained upon the termination of the respective branching procedures. The global lower bound (possibly, the optimal solution value) obtained at the time limit by the setting a is denoted by GLB_a, where $a \in M = \{\texttt{PF+VI}, \texttt{DW+VI}, \texttt{C}, \texttt{AB}\}$. Let $GLB_{\min}$ be the worst global lower bound provided by these methods:

$$GLB_{\min} = \min_{a \in M} GLB_a.$$

We then calculate the relative improvement of the global lower bound with respect to $GLB_{\min}$ as follows:

$$\text{IMP}_a = (GLB_a - GLB_{\min})/GLB_{\min} * 100\%,$$

Figure 5 displays the values of IMP_a across the set of 52 instances for which all tested methods could solve the LP-relaxation at the root node. We notice that for 75% of these instances (i.e., 39 instances) `DW+VI` improves the $GLB_{\min}$ by 5%, the compact formulation improves it by 2.5%, the setting `AB` improves it by 2.1%, whereas `PF+VI` improves it by only 0.6%. Slight differences between the GLB values provided for `C` and `AB` are due to the general purpose cutting planes used by Cplex. For 50 out of 52 instances, the worst bound is improved by up to 10% using `DW+VI`, whereas this improvement is up to 2.5% for `PF+VI`, and up to 5% for `C` and `AB`.

To conclude, the obtained results indicate that the best global lower bounds can be provided by the B&P algorithm derived from the model DW, enhanced by the valid inequalities. However, we emphasize that the full potential of our B&P algorithms has not been exploited with the current implementation, and that it might be unfair to compare our B&P methods against a fully-fledged state-of-the-art MIP solver like Cplex. This is due to the fact that our B&P algorithms cannot benefit from advanced MIP features available at Cplex (as opposed to `C` and `AB`), as we are using our own branching procedure in which Cplex is employed only as an LP-solver.

8. Conclusion

In this paper we have proposed two extended formulations for solving the Virtual Network Functions Placement and Routing problem. The variables of the first formulation (denoted by PF) are latency-constrained paths, whereas the variables of the second formulation (denoted by DW) are latency-constrained paths that also embed the information regarding the function installations at their nodes. In order to strengthen the LP-bounds, we have proposed several families of valid inequalities for both formulations. Their benefits have been computationally demonstrated on a set of instances derived from telecommunication networks. We have presented a branching scheme for each formulation and have developed and implemented the associated Branch-and-Price algorithms. The latter ones are computationally compared with a compact flow-based MIP formulation and a Branch-and-Benders-cut approach (similar to the one from [21]) derived from the compact model by projecting out flow variables.
The obtained results have shown that the (LP-relaxation and global) lower bounds of the compact formulation (and its Benders counterpart) can be improved by both extended formulations. The overall best global lower bounds can be obtained by the model DW, however its CPU time is sacrificed by the expensive MIP-based pricing procedure. Hence, the model DW suffers from a trade-off between the quality of its bounds and the time needed to solve the LP-relaxations.
We conclude that the full potential of the extended formulations is still to be exploited. We believe that there is a theoretical and empirical interest in further studying these models. For example, developing some more advanced exact and heuristic pricing schemes could help to resolve the trade-off between the quality of obtained bounds and the computational difficulty of the pricing problems. Moreover, there is an interest in studying alternative branching schemes, or more advanced heuristics for calculating incumbent solutions, in order to reduce the size of the branching tree.

References

[1] Addis, B., Belabed, D., Bouet, M., & Secci, S. (2015). Virtual network functions placement and routing optimization. In *4th International Conference on Cloud Networking (CloudNet)* (pp. 171–177). IEEE.

[2] Addis, B., Carello, G., & Gao, M. (2020). On a virtual network functions placement and routing problem: Some properties and a comparison of two formulations. *Networks*, *75*, 158–182.

[3] Allybokus, Z., Perrot, N., Leguay, J., Maggi, L., & Gourdin, E. (2018). Virtual function placement for service chaining with partial orders and anti-affinity rules. *Networks*, *71*, 97–106.

[4] Bonami, P., Salvagnin, D., & Tramontani, A. (2020). Implementing Automatic Benders Decomposition in a Modern MIP Solver. In *International Conference on Integer Programming and Combinatorial Optimization* (pp. 78–90). Springer.

[5] Bouet, M., Leguay, J., Combe, T., & Conan, V. (2015). Cost-based placement of vDPI functions in NFV infrastructures. *International Journal of Network Management*, *25*, 490–506.

[6] Chiosi, M., Clarke, D., Willis, P., Reid, A., Feger, J., Bugenhagen, M., Khan, W., Fargano, M., Cui, C., Deng, H. et al. (2012). Network functions virtualisation: An introduction, benefits, enablers, challenges and call for action. In *SDN and OpenFlow world congress* (pp. 1–16). Darmstadt-Germany volume 48.

[7] Chitimalla, D., Kondepu, K., Valcarenghi, L., Tornatore, M., & Mukherjee, B. (2017). 5G fronthaul-latency and jitter studies of CPRI over Ethernet. *Journal of Optical Communications and Networking*, *9*, 172–182.

[8] Dror, M. (1994). Note on the complexity of the shortest path models for column generation in VRPTW. *Operations Research*, *42*, 977–978.

[9] EISGI, N. (2014). Network functions virtualisation (nfv); service quality metrics. , .

[10] Furini, F., Malaguti, E., Durán, R. M., Persiani, A., & Toth, P. (2012). A column generation heuristic for the two-dimensional two-staged guillotine cutting stock problem with multiple stock size. *European Journal of Operational Research*, *218*, 251–260.

[11] Gao, M., Addis, B., Bouet, M., & Secci, S. (2018). Optimal orchestration of virtual network functions. *Computer Networks*, *142*, 108–127.

[12] Gérard, M., Clautiaux, F., & Sadykov, R. (2016). Column generation based approaches for a tour scheduling problem with a multi-skill heterogeneous workforce. *European Journal of Operational Research*, *252*, 1019–1030.

[13] Gouareb, R., Friderikos, V., & Aghvami, A. H. (2018). Delay sensitive virtual network function placement and routing. In *2018 25th International Conference on Telecommunications (ICT)* (pp. 394–398). IEEE.

[14] Gupta, A., Habib, M. F., Chowdhury, P., Tornatore, M., & Mukherjee, B. (2015). Joint virtual network function placement and routing of traffic in operator networks. *UC Davis, Davis, CA, USA, Tech. Rep*, .

[15] Hmaity, A., Savi, M., Askari, L., Musumeci, F., Tornatore, M., & Pattavina, A. (2020). Latency-and capacity-aware placement of chained Virtual Network Functions in FMC metro networks. *Optical Switching and Networking*, *35*, 100536.

[16] Huin, N., Jaumard, B., & Giroire, F. (2018). Optimal network service chain provisioning. *IEEE/ACM Transactions on Networking*, *26*, 1320–1333.

[17] Joncour, C., Michel, S., Sadykov, R., Sverdlov, D., & Vanderbeck, F. (2010). Column generation based primal heuristics. *Electronic Notes in Discrete Mathematics*, *36*, 695–702.

[18] Kuo, T.-W., Liou, B.-H., Lin, K. C.-J., & Tsai, M.-J. (2018). Deploying chains of virtual network functions: On the relation between link and server usage. *IEEE/ACM Transactions on Networking (TON)*, *26*, 1562–1576.

[19] Larsen, L. M., Checko, A., & Christiansen, H. L. (2018). A survey of the functional splits proposed for 5G mobile crosshaul networks. *IEEE Communications Surveys & Tutorials*, *21*, 146–172.

[20] Li, X., & Qian, C. (2015). The virtual network function placement problem. In *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)* (pp. 69–70). IEEE.

[21] Ljubić, I., Mouaci, A., Perrot, N., & Gourdin, E. (2021). Benders decomposition for a node-capacitated virtual network function placement and routing problem. *Computers & Operations Research*, *130*, 105227.

[22] Mehraghdam, S., Keller, M., & Karl, H. (2014). Specifying and placing chains of virtual network functions. In *Cloud Networking (CloudNet), 2014 IEEE 3rd International Conference on* (pp. 7–13). IEEE.

[23] Moens, H., & De Turck, F. (2014). VNF-P: A model for efficient placement of virtualized network functions. In *Network and Service Management (CNSM), 2014 10th International Conference on* (pp. 418–423). IEEE.

[24] Mouaci, A. (Sep 23, 2022). SNDLib instances for the Virtual Function Placement and Routing Problem. https://github.com/mouaciahlam/SNDLib_instances.

[25] Mouaci, A., Gourdin, E., Ljubić, I., & Perrot, N. (2020). Virtual Network Functions Placement and Routing Problem: Path formulation. In *2020 IFIP Networking Conference (Networking)* (pp. 55–63).

[26] Quinn, P., & Nadeau, T. (2015). *Problem statement for service function chaining*. Technical Report RFC 7498.

[27] Sallam, G., Gupta, G. R., Li, B., & Ji, B. (2018). Shortest path and maximum flow problems under service function chaining constraints. In *INFOCOM Conference on Computer Communications* (pp. 2132–2140). IEEE.

[28] Savi, M., Tornatore, M., & Verticale, G. (2015). Impact of processing costs on service chain placement in network functions virtualization. In *2015 IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN)* (pp. 191–197). IEEE.

[29] SNDlib (Dec 10, 2019). SNDlib. http://sndlib.zib.de/.

[30] Tashtarian, F., Varasteh, A., Montazerolghaem, A., & Kellerer, W. (2017). Distributed VNF scaling in large-scale datacenters: An ADMM-based approach. In *2017 IEEE 17th International Conference on Communication Technology (ICCT)* (pp. 471–480). IEEE.

[31] Tomassilli, A., Huin, N., Giroire, F., & Jaumard, B. (2018). Resource requirements for reliable service function chaining. In *2018 IEEE International Conference on Communications (ICC)* (pp. 1–7). IEEE.

[32] Vanderbeck, F. (2011). Branching in branch-and-price: a generic scheme. *Mathematical Programming*, *130*, 249–294.

[33] Yen, J. Y. (1971). Finding the k shortest loopless paths in a network. *Management Science*, *17*, 712–716.