A branch-and-cut algorithm for the availability-aware VNF placement problem in virtualized networks

Rafael Colares
Orange Innovation
Châtillon, France
rafael.colares@gmail.com

Amal Benhamiche
Orange Innovation
Châtillon, France
amal.benhamiche@orange.com

Yannick Carlinet
Orange Innovation
Châtillon, France
yannick.carlinet@orange.com

Nancy Perrot
Orange Innovation
Châtillon, France
nancy.perrot@orange.com

ABSTRACT

In this work, we address the problem of optimally placing Virtual Network Functions throughout a 5G network so that a given set of Service Function Chains can achieve high levels of end-to-end availability. We tackle this problem from a combinatorial perspective and propose a probabilistic approach to evaluate the real end-to-end availability of a service. This generates a non-linear character to the problem which is then linearized to derive an original integer programming formulation for it. We also introduce new families of valid inequalities reinforcing the proposed formulation. Based on these inequalities, we derive an efficient branch-and-cut algorithm.

KEYWORDS

Network reliability, combinatorial optimization, valid inequalities

1 INTRODUCTION

Network Function Virtualization (NFV) is one of the key enabler technologies for tackling the challenges of the upcoming 5G use-cases requirements. These high-level-requirement use-cases include autonomous vehicles, smart factories, smart cities, e-health, for instance. With virtualization, Network Functions gain the ability to be run as applications in Virtual Machines (VMs) or containers on off-the-shelf hardware. This allows higher scalability, more flexibility and reduces network management costs.

Virtual Network Functions (VNF) however are more prone to errors and failures when compared to purpose-built hardware [10, 14, 16]. Indeed, a major challenge for NFV is to ensure high availability levels for its services. The service availability refers to its probability of being operational when required and is defined as the ratio between its expected uptime and total time values (see [2]). A service in NFV-based networks – also called a Service Function Chain (SFC) – is an origin-destination traffic demand composed of a set of VNFs that must be visited in a given order along its route.

In this sense, an SFC is available if and only if all its VNFs can be properly processed. Strict Service Level Agreements (SLAs) impose that SFCs should be highly available (in some cases, for more than 99.999% of the time, which roughly translates to 25.9 s downtime per month [11, 15]). Then, backup VNFs must be placed on the network so that the services can still be ensured even if some network nodes fail.

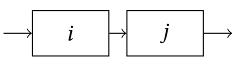
Most of the literature related to SFC's resilience is devoted to securing the network against single-node failures. In this case, it suffices to find two node-disjoint routes (a nominal and a backup) for each SFC. In [4], nominal SFC routes are known in advance and the goal is to find a minimum cost backup VNF placement. In [13], the authors impose a single route for each SFC and focus on minimizing the number of SFCs affected by a single node failure.

A few papers deal with multi-node failures. In [8, 9], heuristic methods treating the multi-node failure scenario are presented. These heuristics (i) construct a VNF assignment for each SFC based on node's remaining resources, (ii) reinforce the assignments by sequentially adding backup VNFs until the availability requirements are met, and (iii) routing is then done through k-shortest paths computations. An interesting exact approach is presented in [5], where service availabilities are computed using a probabilistic approach. However, each service is supposed to request only one VNF which reduces the end-to-end availability computation complexity. In [16], an in-between approach is considered where the goal is to protect the network against single-node failures while knowing that the failure of a node may impact other nodes (failure events are not independent and are related to the network topology structure).

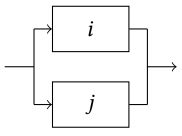
In this paper, we further explore the service availability definition considered in [5] with the purpose of providing a mathematical model that optimizes VNF placements while formally taking into account the SFCs' availability. This allows us to ensure the required SLAs within a multi-node failure scenario. The paper is organized as follows. The problem is formally defined in Section 2 and its computational complexity is briefly discussed in Section 3. In Section 4, we propose an original ILP formulation for the problem which is then reinforced with valid inequalities in Section 5. To verify the efficiency of the proposed inequalities, Section 6 is devoted to the description of a branch-and-cut framework based on such inequalities and Section 7 presents the preliminary computational results obtained with this approach.

2 PROBLEM DEFINITION

Let $G = (V, A)$ be a directed, loopless, connected graph. Each node $v \in V$ has a capacity $C_v \in \mathbb{R}^+$, and an availability $0 < a_v < 1$, (*i.e.*,



(a) Serial composition.



(b) Parallel composition.

Figure 1: Basic compositions of components

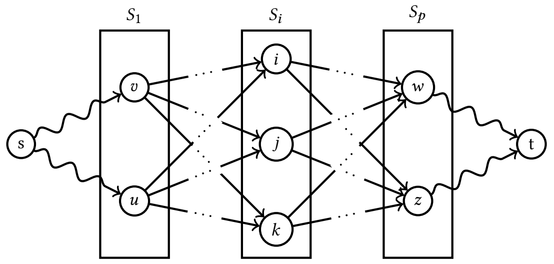


Figure 2: Illustration of a VNF assignment and the possible SFC routes induced by it.

a risk of $1 - a_v$ of being down). Moreover, let $\mathcal{F}$ be the set of VNF types, where each VNF $f \in \mathcal{F}$ has a resource consumption $r^f \in \mathbb{R}^+$, and a placement cost $c_v^f \in \mathbb{R}^+$ for each node $v \in V$. Finally, let K be the set of SFC demands, where each demand $k \in K$ is defined by (i) an origin $o_k \in V$ and a destination $d_k \in V$, (ii) a bandwidth $b_k \in \mathbb{R}^+$, (iii) a required availability $A^k \in [0, 1]$, and (iv) an ordered set of distinct VNFs $F^k \subseteq \mathcal{F}$ that should be visited.

While different VNFs should be placed in series along the SFC route (see Figure 1a), redundant VNFs must be placed in parallel (see Figure 1b) so that whenever one fails, the SFC can be rerouted through one of the redundancies. In serial composition, all subcomponents need to be operational for the system to be available. For a parallel composition, however, it suffices that one subcomponent is operational [12]. It follows that given two subcomponents i and j with respective availabilities a_i and a_j, the availability induced by their serial (resp. parallel) composition is $a_i a_j$ (resp. $1 - \left[(1 - a_i)(1 - a_j)\right]$). Based on these remarks, we next describe the VNF assignment of an SFC as well as the availability it induces.

Given an SFC requiring the visit of p distinct VNFs, let $\mathcal{S} = \{S_1, \dots, S_p\}$ denote a VNF assignment for such SFC, where each nonempty subset $S_i \subseteq V$, for $i = 1, \dots, p$, represents the set of nodes where its i-th VNF can be processed. Figure 2 illustrates such assignment. The subset $S_i \in \mathcal{S}$ is also called the i-th *section* of an SFC. From now on, let $I^k = \{1, \dots, |F^k|\}$ denote the set of sections associated with SFC k. The probability that the i-th VNF of the considered SFC can be properly processed (*i.e.* the availability of its i-th section) is the probability that at least one of the nodes in S_i is operational. This probability, denoted by $a(S_i)$, is hence defined by

$$a(S_i) = 1 - \prod_{v \in S_i} \left(1 - a_v\right). \tag{1}$$

For an SFC to be operational, all its VNFs must be properly processed. The end-to-end SFC availability induced by a VNF assignment $\mathcal{S}$ is denoted by $A(\mathcal{S})$ and given by

$$A(\mathcal{S}) = \prod_{S \in \mathcal{S}} a(S) = \prod_{S \in \mathcal{S}} \left(1 - \prod_{v \in S} \left(1 - a_v\right)\right). \tag{2}$$

A VNF assignment $\mathcal{S}$ is said to satisfy the availability requirements of an SFC k if and only if $A(\mathcal{S}) \geq A^k$.

Given a VNF assignment $\mathcal{S}^k$ for each $k \in K$, let $\mathcal{G} = \{\mathcal{S}^k : k \in K\}$ denote the associated *global VNF assignment.* A global VNF assignment $\mathcal{G}$ is said to be feasible if the availability requirements of each SFC $k \in K$ is satisfied and the amount of resources consumed on any given node $v \in V$ is at most the node's capacity C_v, that is,

$$\sum_{k \in K, i \in I^k : v \in S_i^k} b_k r^{f(i,k)} \leq C_v \qquad \forall v \in V,$$

where $f(i,k)$ is a function mapping the i-th element of F^k, that is, the i-th VNF of SFC k.

Notice that a node $v \in V$ can only process a given VNF $f \in \mathcal{F}$ if f is *placed* on such node. Therefore, given a global VNF assignement $\mathcal{G}$, let $V_f(\mathcal{G}) \subseteq V$ denote, for any $f \in \mathcal{F}$, the set of nodes where f must be placed, that is,

$$V_f(\mathcal{G}) = \bigcup_{k \in K} \bigcup_{\substack{i \in I^k: \\ f(i,k) = f}} S_i^k.$$

The cost induced by a global VNF assignment $\mathcal{G}$ is therefore

$$\sum_{f \in \mathcal{F}} \sum_{v \in V_f(\mathcal{G})} c_v^f,$$

and our goal is to find the feasible global VNF assignment inducing the minimum cost.

3 COMPUTATIONAL COMPLEXITY

Even without taking into account the availability restrictions, the problem is already NP-Hard. Indeed, if each SFC requires the same and only one VNF, then the problem reduces to choosing a minimum cost subset of nodes $S \subseteq V$ where the VNF should be placed such that S is able to treat all SFCs. This is equivalent to a Variable Cost and Size Bin-Packing Problem[1] (see [6, 7]) where bins and items correspond to nodes and SFCs, respectively.

Furthermore, dealing with availability restrictions on their own is also an NP-Hard problem. Indeed, if node capacities are said to be unlimited and there is only one SFC (*i.e.*, $K = \{k\}$) requiring a single VNF to be considered, then the problem reduces to finding a minimum cost subset of nodes $S \subseteq V$ where the VNF should be placed such that the SFC required availability is achieved, *i.e.*, $a(S) \geq A^k$. This is clearly equivalent to a (Nonlinear) Knapsack Problem (see [3]) where each item corresponds to a node in V.

4 ILP FORMULATION

In this section, a natural formulation for the problem is described. The binary variables x_{vik} indicate whether or not the i-th VNF of SFC k can be processed on node v (*i.e.*, if $x_{vik} = 1$ then $v \in S_i^k$, otherwise $v \notin S_i^k$). For each node $v \in V$ and VNF $f \in \mathcal{F}$, the variable y_v^f indicates whether or not VNF f is placed on node v.

$$\min \sum_{v \in V} \sum_{f \in \mathcal{F}} c_v^f y_v^f \tag{3}$$

subject to

[1]The Variable Cost and Size Bin-Packing Problem is a generalization from the well-known Bin-Packing Problem where each bin has its own capacity and cost.

$$\sum_{v\in V} x_{vik} \geq 1 \qquad \forall k \in K, i \in I^k, \quad (4)$$

$$x_{vik} \leq y_v^{f(i,k)} \qquad \forall k \in K, i \in I^k, v \in V, \quad (5)$$

$$\sum_{k\in K}\sum_{i\in I^k} b_k r^{f(i,k)} x_{vik} \leq C_v \qquad \forall v \in V, \quad (6)$$

$$\prod_{i\in I^k}\left(1-\prod_{v\in V}\left(1-a_v x_{vik}\right)\right) \geq A^k \qquad \forall k \in K, \quad (7)$$

$$x_{vik} \in \{0,1\} \qquad \forall v \in V, k \in K, i \in I^k, \quad (8)$$

$$y_v^f \in \{0,1\} \qquad \forall v \in V, f \in \mathcal{F}. \quad (9)$$

The objective function (3) evaluates the total VNF placement cost. The assignment constraints (4) ensure that at least one VNF should be assigned to each section of each SFC. The VNF placement constraints (5) impose that a VNF can only be assigned to an SFC if it is already placed. The capacity constraints (6) guarantee that the capacity of each node is not exceeded. Availability constraints (7) force the SFCs' Service Level Agreement to be respected. Finally, constraints (8) and (9) settle the domains of variables.

The presence of constraints (7) however clearly makes the proposed formulation non-linear. Such non-linearity is difficult to be handled by traditional commercial solvers (*e.g.*, CPLEX, Gurobi). Thereby we next propose a linear reformulation of such constraints. For this, consider the following set of inequalities.

$$\sum_{(v,i)\in(V\times I^k)\setminus\mathcal{S}} x_{vik} \geq 1 \quad \forall k \in K, \mathcal{S} \subseteq V \times I^k : A(\mathcal{S}) < A^k. \quad (10)$$

Proposition 4.1. *An integer solution $\bar{x}$ satisfies inequalities* (7) *if and only if it satisfies inequalities* (10).

Proof. Let $\mathcal{G} = \{\mathcal{S}^k : k \in K\}$ denotes the global VNF assignment associated with solution $\bar{x}$, where $\mathcal{S}^k = \{(v,i) : \bar{x}_{vik} = 1\}$. Let $\bar{x}$ be an integer solution satisfying inequalities (7). Then, $A(\mathcal{S}^k) \geq A^k$ for any $k \in K$. We next show that inequalities (10) are all satisfied by $\bar{x}$. For this, suppose there exists $k' \in K$ and $\mathcal{S}' \subseteq V \times I^{k'}$ for which

$$\sum_{(v,i)\in(V\times I^{k'})\setminus\mathcal{S}'} \bar{x}_{vik'} = 0.$$

By definition, $\mathcal{S}^k \subseteq \mathcal{S}'$ and hence $A(\mathcal{S}') \geq A(\mathcal{S}^k) \geq A^k$, which concludes the first part of the proof. To finish the proof, suppose now that $\bar{x}$ does not satisfy inequalities (7). Then, there exists $k' \in K$ for which $A(\mathcal{S}^{k'}) < A^{k'}$. It follows that, by definition, inequality (10) associated with k' and $\mathcal{S}^{k'}$ is violated. □

Proposition 4.1 shows that inequalities (10) are sufficient for ensuring the required availabilities and hence, we can now replace the non-linear inequalities (7) for (10). The resulting formulation – defined by (3)-(6),(8)-(10) – is linear but also non-compact since it requires an exponential number of constraints. A standard approach is hence to relax such constraints and append them when violated. For this, one needs to deal with the separation problem associated with inequalities (10). Recall that for a family of valid inequalities $\mathcal{I}$, the separation problem for $\mathcal{I}$ consists of either finding an inequality in $\mathcal{I}$ violated by a given vector $(\bar{x},\bar{y})$ or proving that $(\bar{x},\bar{y})$ satisfies all the inequalities in $\mathcal{I}$. For inequalities (10), the associated separation problem amounts to solve a (non-linear) Knapsack Problem (*c.f.* Section 3), that is, an NP-Hard problem.

Proposition 4.2. *The separation problem for inequalities* (10) *can be solved in linear time when vector* $(\bar{x},\bar{y})$ *is integer.*

Proof. Notice that every component of vector $(\bar{x},\bar{y})$ is binary. For each $k \in K$, let $\mathcal{S}^k = \{(v,i) : \bar{x}_{vik} = 1\}$. There exists an inequality in (10) violated by $(\bar{x},\bar{y})$ if and only if there exists $k \in K$ such that $A(\mathcal{S}^k) < A^k$, which can be checked in linear time. □

As a consequence, we propose to solve the associated separation problem heuristically[2] whenever the solution is fractional and exactly otherwise. Nevertheless, such an approach leads to performance issues and the reasons are twofold:

(1) The initially relaxed constraints are the only ones enforcing the assignment of backup VNFs (which directly impacts the objective function). Since their separation problem is only solved exactly on integer solutions, the dual bound convergence is significantly slowed down.
(2) Even if inequalities (10) well-define the availability requirements, they are not strong inequalities. Indeed, the inclusion of a violated inequality (10) imposes that one non-assigned VNF should be in the solution. Such information is quite vague and hence the inclusion of a huge number of inequalities is required to obtain a feasible solution.

The next section is thus dedicated to the reinforcement of the studied formulation through the investigation of valid inequalities.

5 FORMULATION STRENGTHENING

As briefly discussed in the previous section, a major challenge consists of providing good bounds on the number of VNFs required to secure a given SFC without having to appeal to the linearized availability constraints (10). In order to derive such bounds, let us consider the following related combinatorial problem.

Given a set of nodes V and an SFC composed of p VNF types, find the VNF assignment $\mathcal{S}^*$ that induces the highest possible availability for such SFC while placing exactly $n \in \mathbb{N}$ VNFs ($n \geq p$) over the node-set V. We next show that this combinatorial problem may be solved in polynomial time and we use it to derive valid inequalities reinforcing the previously considered formulation.

Claim 1. *If $\mathcal{S} = \{S_1, \ldots, S_p\}$ is an optimal assignment, then each subset S_i is composed of the $|S_i|$ most available nodes.*

Claim 2. *If $\mathcal{S} = \{S_1, \ldots, S_p\}$ is an optimal assignment, then every assignment built from a permutation of sets $S_1, \ldots, S_p$ is also optimal.*

Proof. The availability function $A(\mathcal{S})$ defined by (2) is commutative over the elements of $\mathcal{S}$. □

Claim 3. *If $\mathcal{S} = \{S_1, \ldots, S_p\}$ is an optimal assignment, then the difference between the number of nodes in any two sets S_i and S_j within $\mathcal{S}$ is at most one, that is, $|S_i| - |S_j| \leq 1$.*

Proof. The proof is done by contradiction. Suppose that $\mathcal{S} = \{S_1, \ldots, S_p\}$ is an optimal assignment where there exists i and j for which $|S_i| - |S_j| \geq 2$. From Claim 1, S_i and S_j are composed of the

[2]Our heuristic procedure greedily constructs a VNF assignment $\mathcal{S}$ for each SFC k by iteratively picking the pair (v,i) that maximizes $\bar{x}_{vik}$ and minimizes $A(\mathcal{S} \cup (v,i))$.

most available nodes and hence $S_j \subset S_i$. Let $u_i \in S_i$ be the least available node in S_i. By definition, $u_i \notin S_j$.

Consider the VNF assignment $\mathcal{S}^* = \{S_1^*, \ldots, S_p^*\}$, where $S_k^* = S_k$ for any $k \in \{1, \ldots, m\} \setminus \{i, j\}$, $S_j^* = S_j \cup u_i$ and $S_i^* = S_i \setminus u_i$. Since u_i is the least available node in S_i, we have $a(S_i) > a(S_i^*) > a(S_j^*) > a(S_j)$. Moreover,

$$a(S_i^*) = 1 - \frac{\prod_{v \in S_i}(1 - a_v)}{1 - a_{u_i}} = \frac{a(S_i) - a_{u_i}}{1 - a_{u_i}},$$

and

$$a(S_j^*) = 1 - \left[\left(\prod_{v \in S_j}(1 - a_v)\right)(1 - a_{u_i})\right] = a(S_j) + a_{u_i}(1 - a(S_j)).$$

Next we show that $A(\mathcal{S}^*) > A(\mathcal{S})$, a contradiction since $\mathcal{S}$ is optimal. By definition,

$$A(\mathcal{S}^*) = A(\mathcal{S}) \frac{a(S_j^*) a(S_i^*)}{a(S_j) a(S_i)} = A(\mathcal{S}) \left(1 + \frac{a_{u_i}(a(S_i) - a(S_j^*))}{a(S_j) a(S_i)(1 - a_{u_i})}\right).$$

Since $a(S_i) > a(S_j^*)$, we have $A(\mathcal{S}^*) > A(\mathcal{S})$. □

As a result of Claims 1, 2 and 3, a simple greedy algorithm – see Algorithm 1 – solves the previously presented combinatorial problem.

Algorithm 1: Computation of VNF assignment $\mathcal{S}^*$ inducing the highest possible availability with exactly n VNFs

```
Let $S_i^* = \emptyset$ for $i = 1, \ldots, p$ ;
for $j = 0, \ldots, n - 1$ do
    $i \leftarrow j \bmod p + 1$;
    Add the most available node in $V \setminus S_i^*$ to $S_i^*$;
end
Return $\mathcal{S}^* = \{S_1^*, \ldots, S_p^*\}$ ;
```

5.1 Valid inequalities

Remark 1. *If $\mathcal{S}$ is a VNF assignment such that $A(\mathcal{S}) \geq B$, then $A(\mathcal{S}') \geq B$ for any $\mathcal{S}' \subseteq \mathcal{S}$, since from equations* (1) *and* (2) *we have that $A(\mathcal{S}') \geq A(\mathcal{S})$.*

We next combine the results obtained from Remark 1 and Algorithm 1 in order to derive reinforcing valid inequalities. For this, let $\eta(U, p, B) \in \mathbb{N}$ denote the minimum number of VNFs required to be installed within node-set $U \subseteq V$ so that an SFC composed of p sections can meet the availability requirement B. Notice that $\eta(U, p, B)$ can be easily computed in polynomial time with the help of Algorithm 1. From the definition of $\eta(U, p, B)$, the following inequalities are obviously valid.

$$\sum_{i \in I^k} \sum_{v \in V} x_{vik} \geq \eta(V, |I^k|, A^k) \qquad \forall k \in K.$$

Such inequalities provide a lower bound on the number of VNFs required to be assigned to a given SFC. This can be further extended using Remark 1, which gives rise to the following *Chain Cover* inequalities.

$$\sum_{i \in Q} \sum_{v \in V} x_{vik} \geq \eta(V, |Q|, A^k) \qquad \forall k \in K, Q \subseteq I^k. \qquad (11)$$

Proposition 5.1. *The Chain Cover inequalities* (11) *are valid.*

Proof. From Remark 1, the VNF assignment associated with sections in Q must induce an availability of at least A^k. Thus, inequalities (11) are clearly valid from the definition of $\eta(V, |Q|, A^k)$. □

Chain Cover inequalities provide important information concerning the minimum number of VNFs to be assigned over the sections of an SFC. However, such a number is constructed following the principles of Algorithm 1, that is, only the most available nodes in V are actually taken into account. If for any reason (*e.g.*, elevated cost, insufficient capacity, etc) some of these nodes are banned from being used, such lower bound might increase. We next focus on this case to derive a new family of valid inequalities. For this, consider the following *Node Cover* inequalities.

$$\sum_{v \in V \setminus U} \eta(U, 1, A^k) x_{vik} + \sum_{v \in U} x_{vik} \geq \eta(U, 1, A^k) \qquad \forall k \in K, i \in I^k, U \subseteq V. \qquad (12)$$

Proposition 5.2. *The Node Cover inequalities* (12) *are valid.*

Proof. Let S denote the set of nodes where the i-th VNF of SFC k can be processed for an arbitrary feasible solution. If there is a node $v \in S$ such that $v \in V \setminus U$, then the inequality is clearly satisfied. Hence, suppose $S \subseteq U$. In this case, we have that $|S| \geq \eta(U, 1, A^k)$ from Remark 1 and thus the inequality is also verified. □

Notice that inequalities (12) might become quite loose when there exists a node $v \in V \setminus U$ that can process the i-th VNF of SFC k. For this reason, we next propose a lifted version of such inequalities. For this, let $\beta(v', V, B)$ denote the minimum number of backup VNF replicas that need to be assigned in parallel to v' within the node-set V so that a certain availability requirement B is reached. Remark that such number can be easily obtained by slightly modifying the computation of $\eta(V, 1, B)$. Indeed, it suffices to oblige node v' to be part of the VNF assignment in Algorithm 1. Moreover, let $\bar{V}(v)$ denote the subset of nodes in V that are at most as available as v, that is,

$$\bar{V}(v) = \Big\{u \in V : a(u) \leq a(v)\Big\}. \qquad (13)$$

The *Lifted Node Cover* inequalities are defined as follows.

$$\sum_{v \in V \setminus U} c_v x_{vik} + \sum_{v \in U} x_{vik} \geq \eta(U, 1, A^k) \ \forall k \in K, i \in I^k, U \subseteq V, \qquad (14)$$

where $c_v = \max\Big(\eta(U, 1, A^k) - \beta(v, U \cup \bar{V}(v), A^k), 1\Big)$.

Proposition 5.3. *Lifted Node Cover inequalities* (14) *are valid.*

Proof. Let S denote the set of nodes where the i-th VNF of SFC k can be processed for an arbitrary feasible solution. That is, $S = \{v \in V : x_{vik} = 1\}$. If $S \subseteq U$, then the inequality is satisfied since inequalities (12) are valid. Hence, let us focus on the case where $S \not\subseteq U$. Let v' denote the most available node in $S \setminus U$. By definition, $S \subseteq U \cup \bar{V}(v')$ and hence $|S \setminus v'| \geq \beta(v', U \cup \bar{V}(v'), A^k)$. Since $c_{v'} \geq \eta(U, 1, A^k) - \beta(v, U \cup U_v, A^k)$ and every other coefficient is at least 1, the inequality is verified. □

Chain Cover inequalities (11) deal with the non-linearity of the availability function arising from the chaining of VNFs. Node Cover inequalities (12) treat the availability non-linearity appearing rather from the parallel assignment of VNFs. These two ideas are next combined to form a single large family of valid inequalities. The Generalized Cover inequalities are defined as follows.

$$\sum_{i \in Q} \sum_{v \in V \setminus U} \eta(U, |Q|, A^k) x_{vik} + \sum_{i \in Q} \sum_{v \in U} x_{vik} \geq \eta(U, |Q|, A^k) \qquad \forall k \in K, Q \subseteq I^k, U \subseteq V. \quad (15)$$

Proposition 5.4. *Generalized Cover inequalities* (15) *are valid.*

Proof. Let $\mathcal{S} = \{S_1, \dots, S_{|I^k|}\}$ denote the VNF assignment of SFC k for an arbitrary feasible solution. That is, $S_i = \{v \in V : x_{vik} = 1\}$, for any $i \in I^k$. If there exists a node $v \in S_i$, for any $i \in Q$, such that $v \in V \setminus U$, then the inequality is clearly satisfied. Hence, suppose $S_i \subseteq U$, for every $i \in Q$. In this case, at least $\eta(U, |Q|, A^k)$ VNFs must be assigned to SFC k and hence $\sum_{i \in Q} \sum_{v \in U} x_{vik} \geq \eta(U, |Q|, A^k)$. The inequality is thus verified. □

Next, consider the following *Section Failure* inequalities.

$$\sum_{v \in V} \Big(\log(1 - a_v) \Big) x_{vik} \leq \log(1 - A^k) \qquad \forall k \in K, i \in I^k. \quad (16)$$

Proposition 5.5. *The Section Failure inequalities* (16) *are valid.*

Proof. Let $\mathcal{S} = \{S_1, \dots, S_{|I^k|}\}$ denote a feasible VNF assignment for an SFC $k \in K$, that is, $A(\mathcal{S}) \geq A^k$. It follows directly from Remark 1 that $a(S_i) \geq A^k$ for any $i \in I^k$, *i.e.*,

$$\prod_{v \in S_i} \Big(1 - a_v\Big) \leq 1 - A^k \qquad \forall i \in I^k. \quad (17)$$

Notice that if inequalities (17) hold, then

$$\log\left(\prod_{v \in S_i} \Big(1 - a_v\Big) \right) \leq \log\Big(1 - A^k\Big) \qquad \forall i \in I^k,$$

also hold. Using the fundamental property of logarithms that states $\log(ab) = \log(a) + \log(b)$, such inequalities can be rewritten as

$$\sum_{v \in S_i} \Big(\log(1 - a_v) \Big) \leq \log\Big(1 - A^k\Big) \qquad \forall i \in I^k.$$

Since by definition $x_{vik} = 1$ for $v \in S_i$ and $x_{vik} = 0$ for $v \in V \setminus S_i$, inequalities (16) are hence valid. □

It is worth noting that if one searches for Cover inequalities (see [1]) associated with Section Failure inequalities (16), the inequalities obtained form a subset of the linearized availability constraints (10).

So far, all the introduced valid inequalities have dealt with the availability restrictions. We next propose a family of valid inequalities reinforcing the capacity constraints (6). The *Stronger Capacity* inequalities are defined as follows.

$$\sum_{k \in K, i \in I^k : f(i,k) = f} b_k r^f x_{vik} \leq C_v y_v^f \qquad \forall v \in V, f \in F. \quad (18)$$

Proposition 5.6. *Stronger Capacity inequalities* (18) *are valid.*

Proof. If a VNF is placed on node v, then the amount of resources it consumes must be at most the node's capacity. □

6 BRANCH-AND-CUT FRAMEWORK

We next describe the branch-and-cut framework we have developed based on the results obtained from Section 5. In this framework we consider the linearized formulation (3)-(6),(8)-(10) presented in Section 4 reinforced with Chain Cover inequalities (11), Lifted Node Cover inequalities (14), Section Failure inequalities (16) and Stronger Capacity inequalities (18).

As stated in Section 4, the separation problem for the linearized availability constraints (10) is solved exactly whenever an integer solution is found. This allows us to guarantee the feasibility of the solution provided by the end of the optimization procedure.

Since the Section Failure inequalities (16) and the Stronger Capacity inequalities (18) appear in polynomial numbers, storing them in a pool and checking, by enumeration, whether they all are satisfied remains an efficient way of handling them. For the Chain Cover inequalities (11) and Lifted Node Cover inequalities (14) we next focus on their separation problems.

Proposition 6.1. *The separation problem for the Chain Cover inequalities* (11) *can be solved in polynomial time.*

Proof. For a given SFC $k \in K$, the right-hand side of the inequality depends only on the cardinality of subset $Q \subseteq I^K$. Therefore, one can compute $\eta(V, |Q|, A^k)$, for $|Q| = 1, \dots, |I^k|$, in polynomial time. Additionally, for each chosen cardinality of Q, the left-hand side can be easily minimized by choosing the sections $i \in I^k$ with the smallest values of $\sum_{v \in V} x_{vik}$. □

Solving the separation problem for Lifted Node Cover inequalities (14) is less trivial since the value of the right-hand side depends not only on the cardinality of subset $U \subseteq V$ but also on its composition. For this reason, we consider the following sub-family of inequalities (14) that can be separated in polynomial time by simple enumeration:

$$\sum_{v \in V \setminus \bar{V}(u)} c_v x_{vik} + \sum_{v \in \bar{V}(u)} x_{vik} \geq \eta(\bar{V}(u), 1, A^k), \quad (19)$$

for any $k \in K, i \in I^k, u \in V$, where $\bar{V}(v)$ is defined as in (13) and

$$c_v = \max\Big(\eta(\bar{V}(u), 1, A^k) - \beta(v, \bar{V}(v), A^k), 1\Big).$$

At each node of the branch-and-cut tree, the separation routines are called following a hierarchical order defined by their computational complexity – from the simplest to the hardest – and once a violated inequality is found, the node is re-optimized with the additional cut. All cuts are globally valid. A branching operation is performed once no separation routine can find a violated inequality.

7 COMPUTATIONAL RESULTS

In order to evaluate the efficiency of our approach, this section provides some preliminary results on the computational performances obtained over a small set of randomly generated instances. Two availability scenarios – denoted by *Constant* and *Variable* – were examined on a small network containing 15 nodes. In *Constant* case, all nodes are considered to have the same fixed availability which is set to 0.90. On *Variable*, each node has an availability taken at random between 0.90 and 0.99. For each availability scenario, five sets of $|K| \in \{10, 20, 30, 40\}$ SFCs were randomly generated, where

Table 1: Computational results comparison

Instance		opt		time (s)		gap (%)	
$\|K\|$	Type	(I)	(R)	(I)	(R)	(I)	(R)
10	C	4/5	5/5	0.07	0.04	3.94	0
20	C	3/5	5/5	2.9	1.12	4.35	0
30	C	3/5	5/5	65.5	16.9	15.9	0
40	C	1/5	2/5	2236	540.6	9.48	1.35
10	V	5/5	5/5	3.3	0.2	0	0
20	V	4/5	5/5	95.7	65.8	0.72	0
30	V	0/5	2/5	-	1776	13.3	7.70
40	V	0/5	0/5	-	-	25.3	14.9

each SFC is required to visit between 1 and 5 VNFs chosen at random from a set of 8 VNF types. Moreover, each SFC has a required availability between 0.9999 and 0.999999, which is in accordance with specifications in [15].

Table 1 summarizes the computational results obtained using the initial linearized formulation (3)-(6), (8)-(10) – columns `(I)` – and the reinforced formulation featured in our branch-and-cut framework – columns `(R)`. All our computational experiments were implemented in C++ and performed using the state-of-the-art MIP solver CPLEX 12.10 on a computer equipped with a 1.60 GHz Intel Core i5-8265U processor and 16 Gb RAM. A time limit of one hour was imposed in each run. For each instance size and type (`C` for Constant and `V` for Variable), the number of instances that could be solved to optimality within the time limit is displayed under column `opt`. Column `time (s)` provides the average time in seconds required to achieve optimality. Column `gap (%)` displays the average remaining gap for the instances that could not be solved within time limit.

For the Constant scenario, out of the 20 tested instances, only 11 could be solved to optimality within the time limit by the initial formulation. Our branch-and-cut approach allowed us to solve up to 17 instances to optimality in less than one hour. Indeed, up to 30 demands, all instances could be optimally solved. Moreover, the remaining instances were left with a relatively small gap. For the Variable scenario, the performance of the initial formulation was even worse. Only 9 out of the 20 tested instances could be solved to optimality and none of them had more than 20 demands. With our branch-and-cut framework, 3 more instances could be optimally solved. Besides, the average remaining gaps for the unsolved instances were considerably reduced.

The gain of performance observed with our branch-and-cut approach is largely due to its capability of rejecting unfeasible solutions earlier in the optimization. Indeed, the average number of linearized availability constraints added by the exact separation problem (and hence later in the optimization) dropped from 1440 to 13 in the Constant case and from 1608 to 748 in the Variable case. In addition, considering only the instances that could be solved with both approaches, the branch-and-cut framework was on average 2.92 times faster, and the average number of nodes that were required to be explored in the enumeration tree to prove optimality went down from 148.3 thousand to 38 thousand in the Constant case and from 55.6 thousand to 7.5 thousand in the Variable case.

8 FINAL REMARKS AND NEXT STEPS

In this work, we have studied the problem of optimally placing VNFs throughout a given network so that a set of SFCs can achieve their required availability. The non-linearity inherent to the definition of the end-to-end availability of an SFC represents a major challenge in such problem. We have proposed an original ILP formulation that solves the addressed optimization problem. Such ILP was then reinforced through the investigation of valid inequalities and preliminary computational results testify in favor of their efficiency. Even with the improvements proposed in this paper, our approach can only solve instances of limited size. With this in mind, the use of heuristics can help improve the primal bounds faster and hence speed up the convergence towards the optimal solution. The next steps may also include the consideration of SFCs' routing aspect explicitly into the formulation so that maximum SFC delay constraints and link capacity constraints can be imposed in order to treat a more generalized problem.

ACKNOWLEDGMENTS

This work is supported by the french Agence Nationale de la Recherche (ANR), Project MAESTRO-5G ANR-18-CE25-0012.

REFERENCES

[1] Egon Balas. 1975. Facets of the knapsack polytope. *Mathematical programming* 8, 1 (1975), 146–164.
[2] Eric Bauer and Randee Adams. 2012. *Reliability and availability of cloud computing.* John Wiley & Sons.
[3] Kurt M Bretthauer and Bala Shetty. 2002. The nonlinear knapsack problem–algorithms and applications. *European Journal of Operational Research* 138, 3 (2002), 459–472.
[4] Yannick Carlinet, Nancy Perrot, and Anderson Alves-Tzitas. 2019. Minimum-Cost Virtual Network Function Resilience. In *INOC 2019.*
[5] Marco Casazza, Pierre Fouilhoux, Mathieu Bouet, and Stefano Secci. 2017. Securing virtual network function placement with high availability guarantees. In *2017 IFIP Networking Conference (IFIP Networking) and Workshops.* IEEE, 1–9.
[6] Isabel Correia, Luís Gouveia, and Francisco Saldanha-da Gama. 2008. Solving the variable size bin packing problem with discretized formulations. *Computers & Operations Research* 35, 6 (2008), 2103–2113.
[7] Teodor Gabriel Crainic, Guido Perboli, Walter Rei, and Roberto Tadei. 2011. Efficient lower bounds and heuristics for the variable cost and size bin packing problem. *Computers & Operations Research* 38, 11 (2011), 1474–1482.
[8] Weiran Ding, Hongfang Yu, and Shouxi Luo. 2017. Enhancing the reliability of services in NFV with the cost-efficient redundancy scheme. In *2017 IEEE international conference on communications (ICC).* IEEE, 1–6.
[9] Jingyuan Fan, Zilong Ye, Chaowen Guan, Xiujiao Gao, Kui Ren, and Chunming Qiao. 2015. GREP: Guaranteeing reliability with enhanced protection in NFV. In *Proceedings of the 2015 ACM SIGCOMM Workshop on Hot Topics in Middleboxes and Network Function Virtualization.* 13–18.
[10] Bo Han, Vijay Gopalakrishnan, Lusheng Ji, and Seungjoon Lee. 2015. Network function virtualization: Challenges and opportunities for innovations. *IEEE Communications Magazine* 53, 2 (2015), 90–97.
[11] Bo Han, Vijay Gopalakrishnan, Gnanavelkandan Kathirvel, and Aman Shaikh. 2017. On the resiliency of virtual network functions. *IEEE Communications Magazine* 55, 7 (2017), 152–157.
[12] N Isg. 2016. Network functions virtualisation (nfv); reliability; report on models and features for end-to-end reliability. *ETSI GS NFV-REL* 1 (2016), v1.
[13] Purnima Murali Mohan and Mohan Gurusamy. 2019. Resilient VNF placement for service chain embedding in diversified 5G network slices. In *2019 IEEE Global Communications Conference (GLOBECOM).* IEEE, 1–6.
[14] Justine Sherry, Peter Xiang Gao, Soumya Basu, Aurojit Panda, Arvind Krishnamurthy, Christian Maciocco, Maziar Manesh, João Martins, Sylvia Ratnasamy, Luigi Rizzo, et al. 2015. Rollback-recovery for middleboxes. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication.* 227–240.
[15] 3GPP TS 22.261 version 15.7.0 Release 15. 2019. Service requirements for next generation new services and markets. (2019).
[16] Yordanos Tibebu Woldeyohannes, Besmir Tola, and Yuming Jiang. 2019. Towards carrier-grade service provisioning in nfv. In *2019 15th International Conference on the Design of Reliable Communication Networks (DRCN).* IEEE, 130–137.