# Elastic Network Service Provisioning with VNF Auctioning

Mathis Obadia*†, Mathieu Bouet*, Vania Conan*, Luigi Iannone†, Jean-Louis Rougier†
*Thales Communications & Security
{firstname.name}@thalesgroup.com
†Telecom ParisTech, LTCI
{firstname.name}@telecom-paristech.fr

***Abstract*—Network Function Virtualization (NFV) is an emerging approach that has received attention from both academia and industry as a way to improve flexibility, efficiency, and manageability of networks. NFV enables new ways to operate networks and to provide composite network services, opening the path toward new business models. As in cloud computing with the Infrastructure as a Service model, clients will be offered the ability to provision and instantiate Virtual Network Functions (VNF) on the NFV infrastructure of the network operators.**

**In this paper, we consider the case where leftover VNF capacities are offered for bid. This approach is particularly interesting for clients to promptly provision resources to absorb peak or unpredictable demands and for operators to increase their revenues. We propose a game theoretic approach, using Multi-Unit Combinatorial Auctions to select the winning clients and the price they pay. Such a formulation allows clients to express their VNF requests according to their specific objectives. We solve this problem with a greedy heuristic and prove that this approximation of economic efficiency is the closest attainable in polynomial time and provides a payment system that motivates bidders to submit their true valuations. Simulation results show that the proposed heuristic achieves a market valuation close to the optimal (less than 10% deviation) and guarantees that an important part of this valuation is paid as revenue to the operator.**

## I. Introduction

The rise of Network Functions Virtualization (NFV) represents a major shift in the way telecommunication networks and services are operated and used. So far, network functions such as Firewall, Deep Packet Inspection (DPI), Intrusion Detection Systems (IDS), Video Encoding, Load Balancing, and Routing were supported by hardware middleboxes from multiple vendors. With the NFV approach [1], these functions can now be virtualized — becoming Virtual Network Functions (VNFs). These individual functions run on commodity hardware and can be chained to form complex offers, thus promising manageability and cost-efficiency. VNFs can be instantiated on Points-of-Presence (PoPs), which already host embedded cloud infrastructure next to legacy middleboxes.

Yet, NFV goes beyond a mere technical change. Network operators can leverage this approach to enrich their offer and develop new business models. We can envision network operators directly selling NFV resources as infrastructure, with clients choosing where to place their VNFs inside such network to better suit their needs. This can be described as Virtual Network Function as a Service [2]. When provisioning their own network services, the clients can estimate with some accuracy their average and peak needs, and how much gain they can expect from satisfying these demands. If they want to always satisfy all demands, over-provisioning is the norm, but the additional cost of reserving resources left mostly unused is not cost efficient. At the same time, the operator has often unsold resources that could be offered to anyone interested, using an auction mechanism with dynamic prices to sell those available resources. In such a system, clients bid on leftover resources, for instance to satisfy their peak and unpredictable demands. In this way, they only satisfy those demands when the competition for those resources is not too aggressive and the price paid remains low. Hence, from client's perspective, the idea is to buy resources to absorb peak demand at low price, instead of over-provisioning. For the network operator, instead, this system can allow to sell resources that would otherwise be unused (generating no income). A similar auctioning system has been successfully used in the cloud environment (Amazon EC2 [3]), providing benefits to both clients and cloud operators [4].

In the NFV setting, instead Virtual Machines, clients want to acquire a set of ordered VNFs linked together called Service Function Chains (SFCs). The problem of mapping SFCs requests with respect to available network resources has been already explored in the literature, e.g., see [5]. In such formulation, to maximize network utility, two main issues need to be addressed: (i) choose which clients to serve when resources are limited (*winner determination*); (ii) accepted services should be placed/allocated in the network infrastructure, meeting potentially complex demands in term of service chaining (*mapping*).

In this paper, we tackle the problem of maximizing network operator utility in economic terms, introducing and addressing a third issue: (iii) decide how much the clients should pay (*payment plan*). We propose a VNF auction which, acting as an interface between the operator and the clients, is tasked to accept or reject clients' demands and fix their price to maximize market value.

With the proposed approach, clients run their own VNF composition and optimization process, taking into account their own constraints and needs and what the operator advertises as available resources, and request VNF function chains. In order to solve the problem of *winner determination* and

978-0-9883045-1-2/16 $31.00 © 2016 ITC

find the *payment plan*, we use Multi-Units Combinatorial Auctions [6] scheme, which allow us to model the demands of a bundle of VNFs (i.e., a VNF chain) and provides: (i) the most economically efficient clients to serve, and (ii) a dynamic pricing for each service function chain as a whole.

We solve this problem with a greedy heuristic which is proved to be the best approximation attainable in a polynomial time for the winner determination problem. We also demonstrate that it provides a payment plan that is incentive compatible and strategy proof. In other words, the goal of the payment plan is to give incentives to clients to submit their true valuations as a bid. We want to avoid clients strategically placing their bids to game the system, as it could result in inefficiencies and instabilities. This is also desirable as it is fair to clients who gain no value from the additional information that they could have about their competitors and their bids [7].

The rest of the paper is structured as follows. Sec. II presents some related work. Sec. III introduces the proposed VNF auction scheme. Sec. IV provides the market model Sec. V presents the different algorithms that can be used to allocate resources and decide the price of each SFC. Those algorithms are evaluated and compared in Sec. VI. Finally, Sec. VII concludes the paper.

## II. Related Work

Some work have proposed approaches to find optimal solutions and heuristics for the VNF placement and chaining problem ([8], [9], [10]). They all look at the problem where demands are expressed as chains with a certain number of constraints and the operator is in charge of both the optimization and the orchestration processes. We propose a system that is similar to the cloud markets, where demands are expressed directly into infrastructural needs to the cloud operator [11].

The proposed cloud market schemes using combinatorial auctions is similar to [12], which uses double sided auctions to sell cloud resources in a marketplace similar to the electricity market, using a combination of spot and future market. Different types of cloud markets and pricing models are reviewed in [13]. Auctions have also been used for bandwidth reservation in networks [14] and can be solved using second price auctions in case where demand is elastic [15]. The main difference between our approach and these markets is that we apply this paradigm to network function embedding in a NFV setting. We want to express the fact that a service function chain is only useful if all the components are successfully bought by the client and implemented by the operator. One way to model such constraints is to use Multi-Unit Combinatorial Auctions.

Combinatorial auctions have been used in a variety of settings from spectrum auctions [16] to airport time slot allocation [17]. With this settings, a variety of single goods are sold in an auction where bidders try to acquire a bundle of goods. Combinatorial auctions are important tools of the so-called "mechanism design" theory, because they are applicable to a wide range of different settings. They still pose some computational challenges, that were addressed in [18]. This work introduces an algorithm for the case of single unit combinatorial auctions. Our approach extends their approach for multi-unit combinatorial auctions, where each goods have a certain limited quantity that can be greater than one, which is more adapted to selling network resources, such as link capacity where we don't sell a full link but bandwidth on this link.

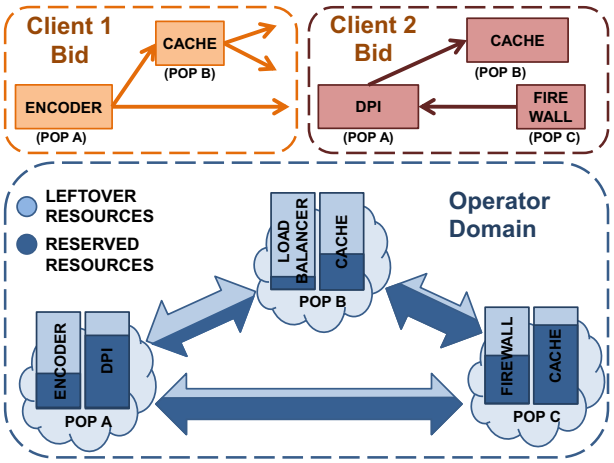


Fig. 1: Content Delivery Networks use-case example.

Works in multi-unit combinatorial auction have focused on finding the optimal allocation (winning bids) which is a computationally intensive task even with adapted heuristics ([19], [20]). Incentive compatible payment systems, with optimal winner determination, have also been found on easier instances of the problem as in [21], where each bidder is limited to buying a quantity of one unit per good.

Our work adapts auctions to the combinatorial setting of NFV and service chaining, which is the main difference from the classical cloud setting.

## III. VNF Auctioning

### A. Motivations

Because of bursty demand, over-provisioning is a necessity, but buying resources at full price in a regular fixed price market for mostly unused resources comes at a high price for clients. Our system aims at offering an alternative method to over-provisioning in order to accommodate unexpected demands, by quickly buying additional resources for a usually smaller price. We can also envision that some of the resources that were bought in the fixed price market because of over-provisioning can then be auctioned if they are left unused. In Fig. 1 we consider a simple use case with two clients who want to deploy parts of a Content Delivery Network. They both compete for the operator resources: the two clients are both interested by the same caching resources located in PoP B and the same link capacity between PoP A and PoP B. In the case where the operator does not have sufficient leftover resources (the lighter (blue) colored part of VNFs in the operator domain box in Fig. 1) to satisfy both clients, it

will need to make a choice based on the bid price submitted by each client. The operator cannot make decisions based on each VNF individually, because a single un-deployed VNF in the service chain makes the whole chain useless: for instance, it is useless to obtain a firewall for client 2 if he does not get the caching resources.

**Clients' motivations:** Provisioning is a complicated process, the clients need to understand the variability of their demands, and the business objectives related to their needs. Each of the client's users is providing a given utility (e.g. the price paid by the users) and his demands can be estimated to buy most of the fixed reserved resources in the network (the darker (blue) colored part of VNFs in the operator domain box in Fig. 1). But part of the demand is variable and unpredictable. Thus, instead of over-provisioning, the client can choose to serve the peak users only if the cost of buying additional capacity does not exceeds the utility of serving them. The client can then use this valuation of serving additional clients as the bid for the desired resources (resources which are displayed in the client bid boxes in orange and red in Fig. 1).

**Operator's motivations:** The operator has spare resources that can be sold to a number of different possible clients. Since clients are only interested in full service chains, the auctioning system needs to let clients express their demands as bundles of resources associated with the bid price associated with the whole chain. The operator benefits from such a common format as it frees him from the need to know what are the most relevant SFCs that should be proposed to clients. The system is based on an auction mechanism: this introduces dynamic pricing with prices paid according to supply and demand, as in cloud auction systems for virtual machines. In this way the operator is able to distribute his limited resources to the clients who value them most (and are thus willing to pay more). The market decides which requests are satisfied and then the whole VNF chain is deployed in the operator's infrastructure. For a client a request is either not satisfied (because the price to pay would be higher than the valuation associated to it), or the entire service function chain is accepted (and the price is below its valuation). This is achieved using Multi-Unit Combinatorial Auctions that take into account that each resource has a maximal capacity and that they are bought in bundles. For example, a client who submitted a bid for a video streaming service chain with 10 mb/s links and a video encoder for $100 might only pay $80 if his bid is accepted and will always pay $0 if it is not accepted.

### B. Market phases

The problem of allocating resources is solved by using a multi-unit combinatorial auction market on a time slot basis. This means that all resources are freed after each time slot and that each time slot is considered as independent one. Service function chains required to span more than one time slot have no guarantees to win all the auctions. This does not introduce any issue in cases like time slot longer than the demanded time (which is the peak duration, in our use case), or when used to treat time shift-able demands. This system is adapted for selling unused residual resources, which were not sold with the traditional (non-auction based) long term plans offered by the operator. The two systems (high price guaranteed duration and lower price auction) need to coexist to offer clients the reliability that their service chain requires. This is similar to a cloud system such as Amazon Spot Instances that can be disconnected any time when the real time price exceeds the maximum target price chosen by the client [22]. In the same way the client is assured that he will always pay less than the bid price and will only pay for the instant dynamic price which is typically quite low but in turn is not offered a guarantee on the duration of the services supplied. We do not make any assumption on the duration of the time slot at this stage, rather focusing in this paper on what happens during each time slot.

To attribute resources for each time slot, the VNF auctioning system can be devised in five consecutive phases:

1) **Resource advertisement phase:** The operator has to advertise the VNFs that it can provide, the virtual topology of his network, e.g., the available PoPs, the different VNF availability within these PoPs and their capacity, and the capacity of the connecting virtual links. This will inform clients about what are the available resources in order for them to build the service function chains they need and can buy.
2) **Bidding phase:** The clients choose bundles of available resources that represent a service function chain. They also submit the bidding price which represents the maximum price they are willing to pay for that bundle.
3) **Winner determination phase:** After the bidding phase, the auction selects which bids are accepted and which bids are refused, because of a lack of resources.
4) **Price computation phase:** The auction will also determine how much each bidder has to pay. We assume that losing bidders pay zero and winning bidders pay a price that is less than or equal to their bidding price.
5) **VNF instantiation phase:** All previous phases have to end before the beginning of a time slot to allow service function chains to be properly instantiated.

## IV. VNF AUCTIONING MODEL

We model the VNF auctioning so as to represent the information exchanged in the system: the offer from the operator and the clients' demands are expressed as bids. Fig. 2 shows an example of a possible mapping between two service function chains from clients' demands and the virtual NFV infrastructure of the operator.

### A. Network operator model - seller

Network operators control the Infrastructure as a Service (IaaS) and sell directly the virtual resources of bandwidth in their links and the VNFs that they want to offer. We use the general term of *service* for both kind of resources:

- **Bandwidth:** the available bandwidth the operator wants to sell, expressed in Mb/s, between ingress, egress nodes and PoPs.

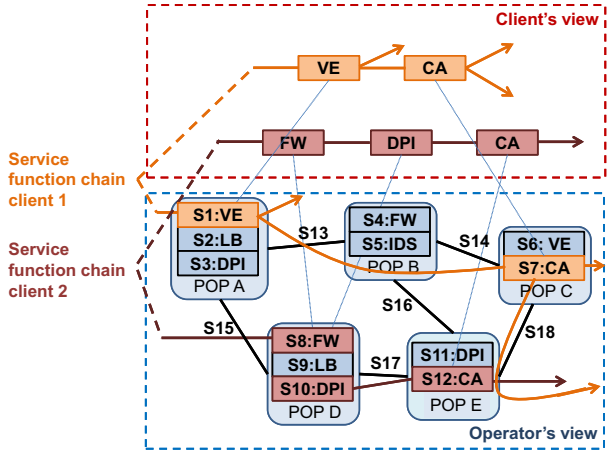



Fig. 2: Example of mapping of different service function chains onto the NFV infrastructure.

- **VNF:** the available VNF functions offered by the operator. Each VNF demand is defined by a quantity of the service expressed in a unit, specified in the service description (most relevant to this particular function). For instance, DPI might be expressed in Mb/s of inspected flows. Firewalling might be expressed in number of flows/s that can be inspected.

Each of the $M$ services $\{S_1, S_2, ..., S_M\}$ sold in the market is specific to a particular PoP or link. For instance, a client can choose to buy a Firewall (FW) VNF in a PoP in New York or in Los Angeles. In Fig. 2 for instance, there are a total of $M = 18$ services offered on 5 PoPs and 6 links ($S_1, S_2, S_3$ in $PoPA$, $S_4, S_5$ in $PoPB$, etc.). The maximal available capacity for service j is $C_{S_j}$.

### B. Client's demand model - bidder

Each client's demand is expressed as an abstract service function chain with some requirements on how this chain is embedded in the network. For example the service function chain of client 2 in Fig. 2 links three VNFs: Firewall (FW), Deep Packet Inspection (DPI) and Cache (CA). It can expressed as *(<ingress:10 Mb/s>→<FW:10Mb/s>→<link:10Mb/s>→<DPI:10Mb/s>→ <CA:10Mb/s>→<egress:20Mb/s>)*, could be instantiated in a number of different ways. It is up to the client to submit bids that are in accordance with his constraints. The proposed bid in our example is *{<S8:10Mb/s>,<S10:10Mb/s>,<S12:10Go>,<S17:10Mb/s>}*. If a client has multiple bids that satisfy his requirements, he can submit alternative bids: *{<S8:10Mb/s>,<S10:10Mb/s>,<S12:10Go>,<S17:10Mb/s>}* OR *{<S8:10Mb/s>,<S11:10Mb/s>,<S12:10Mb/s>, <S17:10Mb/s>}*. This assures the client to win only a single service function chain, while increasing his chances.

More formally we denote $Q_i = \{q^i_{S_1}, ..., q^i_{S_j}, ..., q^i_{S_M}\}$ the set of services demanded by bidder $i$, where $q^i_{S_j}$ denotes the quantity of service $S_j$ required by bidder $i$. We assume that each bidder $i$ has a valuation $v_i$ for the entire set of services $Q_i$. This means that client $i$ values the service as *zero* if he does not get at least the demanded quantity for each service. A bid is defined as a tuple $(Q_i, b_i)$ where $Q_i$ represents the services demanded and $b_i$ represents its bid price, this price can be chosen strategically by the client to maximize his utility. The number of bidders is expressed by $N$.

TABLE I: Notations.

| | |
|---|---|
| $b_i$ | Bid price submitted by bidder i |
| $C_{S_j}$ | Capacity of Service $S_j$ |
| $M$ | Number of services |
| $N$ | Number of bids |
| $p_i$ | Price paid by bidder i |
| $Q_i = \{q^i_{S_1}, ..., q^i_{S_M}\}$ | Quantities of each services asked by bidder i |
| $q^i_{S_j}$ | Quantity asked by bid i for service $S_j$ |
| $S = \{S_1, ..., S_M\}$ | Services offered |
| $u_i$ | Utility of bidder i |
| $v_i$ | Valuation of bidder i |
| $W$ | Set of accepted bids |
| $x_i$ | Boolean: 1 if i is a winning bid 0 otherwise |

### C. Market's properties

As seen in previous sections, the market needs to perform two tasks: winner determination and price computation. We denote $W$ the set of accepted bids. We denote as $p_i$ the price bidder $i$ has to pay. We assume that losing bids do not pay anything (i.e., $p_i = 0$ if the bid is not accepted). Note that the price $p_i$ is not necessarily the same as his bid price $b_i$. The way $p_i$ is computed has a big influence on the strategy that bidder $i$ will use on choosing his bid price $b_i$. The following are some desirable game theory properties that a market can have, but that cannot be respected all at the same time.

**Incentive compatibility:** the market is made in such a way that the best possible outcome for each participant is to reveal truthfully its private parameters required by the system, i.e. $b_i = v_i, \forall i$ in our case. In our mechanism, the price paid by the winning bidder is always lower or equal to the submitted bid price, thereby restraining the strategic behaviors of clients trying to submit the lowest possible bid that would still allow them to win the desired service function chain. This also means that the price paid by the client does not depend on his own bid price — this price is only used to determine whether he wins or not.

**Maximize operator revenue:** maximizing the payments made by the winning bidders $\sum_{i \in W} p_i$.

**Economic efficiency:** In auction theory, economic efficiency refers to allocating the services to bidders who value it the most. This means maximizing the total sum of all accepted valuations $\sum_{i \in W} v_i$. Another possible objective is to maximize the social welfare, i.e. the global utility of the system, composed of the seller and all the bidders. This represents the utility of the bidders $\sum_{i \in W}(v_i - p_i)$ and the utility (revenue) of the seller $\sum_{i \in W} p_i$. In other words, it represents the total valuation of the accepted bidders $\sum_{i \in W} v_i$.

In this particular case, economic efficiency and social welfare maximization are aligned.

There is a trade-off between revenue and efficiency maximization; it is a well-known problem that has been extensively studied [18]. Nevertheless, keeping efficiency high is very important when considering long-term business benefits. Further justifications of this choice are provided in Sec. V-E. The heuristic we designed follows the first and last properties: incentive compatibility and economic efficiency. It is not possible to maximize operator revenues together with those two properties; however, we will discuss how our system affects operator revenue in the following section. Notations are summarized in Table I.

### D. Dummy Services

In a system where there can be more than one way to implement a service for a client, it is useful for clients to submit alternative bids, as explained in sec. IV-B. The problem of representing alternative bids is that it is easier to solve classic markets bids where only one set has a value for the bidder and all the rest has zero valuation (so-called "single minded bidders" in auction theory). However, in order to allow a more realistic representation of the bidder valuation, it is possible to represent alternative (*OR*) bids with dummy services [23]. Each alternative bids are split into normal bids with an added dummy service with a capacity of one asked by each bid. This ensures that only a single bid (among these alternatives) will be a winning bid. This is because when any of the alternative bid is chosen this consumes all the available units of the dummy service and it is not possible to accept any other. For instance, the bid:*({<S1:10>,<S2:20>}, $100)* OR *({<S3:10>,<S4:20>}, $90)*, by creating the dummy service S5 (with capacity $C_{S_5} = 1$) becomes: bid 1: *({<S1:10>,<S2:20>,<S5:1>}, $100)*, bid 2: *({<S3:10>,<S4:20>,<S5:1>}, $90)*. Note however that this can make the number of goods (i.e. services) grow quickly and that might pose some computational challenges. We prove in Sec. V-D that our heuristic gives a result in polynomial time according to the number of goods so it does not pose a problem for this particular heuristic.

## V. Multi-Unit Combinatorial Auction Heuristic

The problem of finding the optimal economic efficiency (i.e., maximizing the total valuation of winning bids) is proven to be NP-complete [18]. We thus propose a heuristic and detail it in this section.

### A. Economic efficiency

Here we assume that we have an incentive compatible mechanism (as discussed in the previous section). This means that $b_i = v_i$, $\forall i$ and maximizing the valuations of accepted bidders $\sum_{i \in W} v_i$ is equivalent to maximizing the total of accepted bid prices: $\sum_{i \in W} b_i$. The economic efficiency objective can be easily expressed with an Integer Linear Programming (ILP) representation: let $x_i$ the Boolean variable representing whether bid $i$ is accepted or not.

**Algorithm 1** Greedy incentive compatible heuristic

**initialization:** Reorder the bids such that: $\frac{b_1}{\sqrt{\sum_{j=1}^{M} q^1_{S_j}}} \geq \frac{b_2}{\sqrt{\sum_{j=1}^{M} q^2_{S_j}}} \geq ... \geq \frac{b_N}{\sqrt{\sum_{j=1}^{M} q^N_{S_j}}}$
$\forall j \in [1..M] \quad U(j) = 0$ represents how many available units of service j have been used
**Output:** $W$: set of winning bids, $p_1, ..., p_N$: price paid for each bid

```
// Part 1. Winner Determination
for i=1...N do
    if ∀j ∈ [1...M]  q^i_{S_j} + U(j) ≤ C_j then
        W ← i
        ∀j ∈ [1...M]  U(j) = U(j) + q^i_{S_j}
    end if
end for
// Part 2. Price Computation
for i ∈ W do
    ∀j ∈ [1..M]  U(j) = 0
    for k = [1...N], k ≠ i do
        if ∀j ∈ [1..M]  q^k_{S_j} + U(j) ≤ C_j then
            ∀j ∈ [1..M]  U(j) = U(j) + q^k_{S_j}
        end if
        if ∃j such that q^i_{S_j} + U(j) ≥ C_j  then
            p_i = b_k * sqrt(Σ_{j=1}^{M} q^i_{S_j}) / sqrt(Σ_{j=1}^{M} q^k_{S_j})
            break
        end if
    end for
end for
```

**Maximize:**

$$\sum_{i=1}^{N} x_i b_i \tag{1}$$

**Subject to:**

$$\sum_{i=1}^{N} x_i q^i_{S_j} \leq C_j \quad \forall j \in [1..M] \tag{2}$$

This objective corresponds choosing the accepted clients so as to maximize the sum of the accepted bid prices $x_i b_i$ while not accepting clients if there is not enough leftover resources $C_j$ on any link or VNF $j$ to satisfy their demands $q^i_{S_j}$. This objective is solved in our optimal evaluation using the CPLEX linear solver. CPLEX can find the optimal of this optimization problem, however, since it is NP-complete, the time to solve it becomes very long for big instances of the problem.

### B. Greedy heuristic

To alleviate the computation time problem of finding an optimal solution, we propose to use a greedy heuristic. The heuristic is presented in Algorithm 1 and has two main objectives. Firstly, finding an approximation in a polynomial time for the winner determination problem that is solved in the preceding ILP. Secondly, computing the prices paid by each winning bid. This heuristic is a $\sqrt{V}$ approximation of the maximal sum of accepted bid prices [19], where $V$ is the maximal sum (the one found using the ILP). The heuristic aims at solving the two parts in a consecutive manner:

**Winner determination:** In this part (lines 1-7) we try to maximize the sum of the bid prices of all winning bids with a greedy algorithm. We reorder bids proportionally to their bid price and weight inversely with the square root of the sum of the units asked. We then add bids in that order if they are enough service units left until the last bid. This order is chosen because R. Gonen et al. [19] proves that it is the best approximation attainable in a polynomial time for this kind of problems. If $V$ is the maximal sum of accepted bid prices, this will give a $\sqrt{V}$ bid price sum in the worst case scenario.

**Price computation:** In this part (lines 8-20) we determine how much each winning bids should pay (remember that losing bids pay zero). For each winning bidder $i \in W$, ranked $i^{th}$ in the greedy order, we compute the optimal allocation without bid $(Q_i, b_i)$ and, at each step $k$, checking whether satisfying its demand $q^i_{S_j}$ for all $j$ would have been possible. Since $i$ is a winning bidder, the rank $k$ for which his demand cannot be satisfied anymore is greater than $i$. This rank $k$ represents the maximum rank that $i$ could have obtained while still winning the bid. The bid price that $i$ should have had to be at rank $k$ is $p_i = b_k \sqrt{\sum_{j=1}^{M} q^i_{S_j}} / \sqrt{\sum_{j=1}^{M} q^k_{S_j}}$. This is the minimal bid price that bidder $i$ could have used and still win.

### C. Incentive compatibility

**Claim 1.** *The greedy heuristic is incentive compatible.*

*Proof.* Let's denote $u_i$ the utility function of bidder $i$. $u_i = v_i - p_i$ if bidder $i$ wins the desired services $Q_i$ and is zero otherwise. Note that from the way the algorithm is devised $u_i \geq 0$. We want to prove that for any bidder $i$, submitting bid price $b_i \neq v_i$ or services $Q'_i \neq Q_i$ and paying $p'_i$ as a bid will not improve his utility, defined as $u'_i$. Let's define the relation $\subseteq$ for service demands such as: $Q_i \subseteq Q'_i$ if $\forall j \in [1..M] \quad q^i_{S_j} \leq q'^i_{S_j}$. The only way of obtaining all services $Q_i$ for bidder $i$ is if his bid $Q'_i$ is such that $Q_i \subseteq Q'_i$.

$u'_i$ is the utility bidder $i$ gets by submitting $(Q'_i, b_i)$. if $(Q'_i, b_i)$ is not a winning bid, $u'_i = 0$ this cannot increase his utility. We then assume $(Q'_i, b_i)$ is a winning bid. If $Q_i \nsubseteq Q'_i$ bidder does not obtain the desired services but pays $p'_i$. his utility is negative. We then assume $Q_i \subseteq Q'_i$ and $(Q'_i, b_i)$ is a winning bid.

We then show that the bidder will always be better off by reporting $(Q_i, b_i)$ rather than $(Q'_i, b_i)$. Since $Q_i \subseteq Q'_i$ $\sum_{j=1}^{M} q^i_{S_j} \leq \sum_{j=1}^{M} q'^i_{S_j}$ and the bid is placed lower in the greedy order. Since the critical rank $k'$ is the lowest rank that $(Q'_i, b_i)$ could have been and still won the auction, and since $(Q'_i, b_i)$ is asking for more resources than $(Q_i, b_i)$ then $k' \leq k$. This means that $\frac{b'_k}{\sqrt{\sum_{j=1}^{M} q'^k_{S_j}}} \geq \frac{b_k}{\sqrt{\sum_{j=1}^{M} q^k_{S_j}}}$. The critical payment $p'_i$ is then equal to $\frac{b'_k * \sqrt{\sum_{j=1}^{M} q'^i_{S_j}}}{\sqrt{\sum_{j=1}^{M} q'^k_{S_j}}} \geq \frac{b_k * \sqrt{\sum_{j=1}^{M} q^i_{S_j}}}{\sqrt{\sum_{j=1}^{M} q^k_{S_j}}} = p_i$

It is left to show that bidding $(Q_i, v_i)$ is always better or equal than the winning bid $(Q_i, b_i)$. If $(Q_i, v_i)$ is a winning bid then reporting $b_i \geq p_i$ will give the same result and same payment, reporting $b_i \leq p_i$ will result in a losing bid and a worse or equal utility. If $(Q_i, v_i)$ is a losing bid and since $(Q_i, b_i)$ is winning bid it means that the critical payment necessary to win this bid $p'_i$ is greater than the true valuation $v_i$. The utility is then $v_i - p'_i \leq 0$. □

Since the algorithm provides an incentive compatible payment system, the approximation of the winner determination also provides an approximation of economic efficiency (as discussed in Sec. V-A).

As a note we can remark that we proved that the algorithm is strategy proof for a single player. Meaning that no single player has an incentive to not be truthful about his valuation. But we do not make the claim that it is group-strategy proof, meaning that a coalition of bidders can gain from submitting untruthful bids. To see how that is possible consider the scenario where player $i$ and $j$ are in a coalition. If $i$ is a winning bid, the price paid by $i$ depends on the bid submitted by another player. If that other player is $j$ then there can be an incentive for the player $j$ to submit a bid price that is lower than his real valuation because it will lower the cost paid by $i$. But by lowering his bid, player $j$ also has a chance to loose his bid and utility gained. In a competitive environment this scenario should prove to be rare and not overly affect the system as the same issue is present in Amazon EC2 auctions [22].

### D. Computational complexity

The heuristic we use starts by ordering all the bids, if we use the quicksort algorithm we get a complexity of at worst $\mathcal{O}(N^2)$ with N being the number of bids [24]. The algorithm then performs N*M operations to check if the bid can be accepted and for each accepted bid performs N*M operations to find the worst possible position. The complexity then stays at $\mathcal{O}(N^2 * M)$ in the worst case scenario. Having a high number of dummy bids that can make the number of services M grow quickly does not pose a particular challenge for this heuristic.

### E. Effect of incentive compatibility on operator revenue

It is easy to see that the incentive compatible greedy mechanism guaranties economically efficient winning bids, but has no guaranty about operator revenue. An operator could try to maximize his revenue by choosing $p_i = b_i$ for all winning bids, after having maximized the total valuation of accepted bids. However, such mechanism gives an incentive for bidders to report bid prices that are lower than their valuations: they will try to guess what the lowest price they could submit and still win is, using the guess as a bid price. Depending on the conditions and how much risk bidders are willing to take, this auction could give economically inefficient results. It would also give an unfair advantage to clients that have more information about the competition and their respective bid submissions. Having an incentive compatible mechanism makes the task of submitting bid much easier for clients since there is no need to try to guess the bid prices of other clients: their task is limited to set the maximum price they are willing to pay for a service function chain, while being assured that

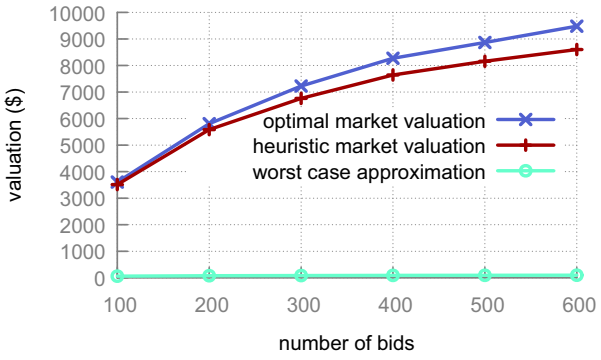



Fig. 3: Difference between the optimal total valuation and the heuristic total valuation with $N_{VNF} = 3$, $C = 100$.

they will pay the minimum price possible (and maximize their own utility).

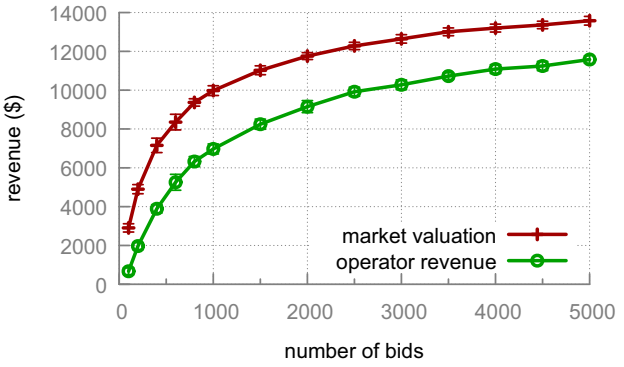
## VI. Evaluation

### A. Implementation and experimental setup

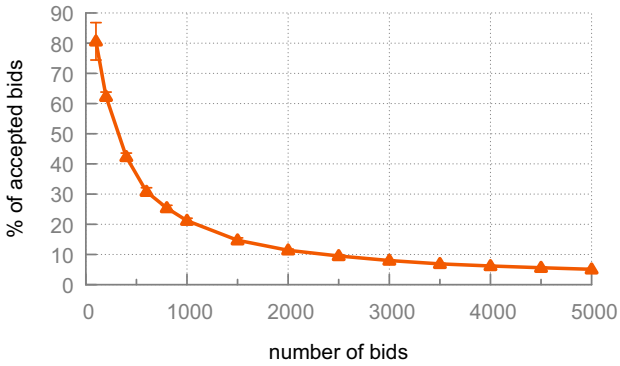
We used a simple model for our experimental setup to show with a few parameters how the whole system reacts when we introduce our NFV auctioning.

**Operator model:** The topology used is the GEANT network with 27 nodes, and is representative of a medium sized backbone network that is already hosting a number of services such as VPN or cloud virtualization. On this network, traffic is highly elastic with total load for 15 minutes varying during a day between 600 GBytes and 1.4 TBytes and some link resources are highly loaded (attaining 100% load) even if the network is mostly over-provisioned and has a mean max load of around 10%. [25]. In our evaluation nodes are considered as PoPs, and the 37 edges as high-capacity links. The services offered are bandwidth on each link and $N_{VNF}$ services on each node. Thus, the number of services is: $M = 37 + N_{VNF} * 27$. The service capacity is $C$ and is the same for all services.

**Bid model:** The number of bids is equal to $N$. To define each bid, we need to model the demanded services (bandwidth and VNFs), their quantity, their valuation and their bid prices. Each bid represents a service function chain going from a random ingress node to a random egress node. Demanded services are: *(i)* bandwidth on all the links of the shortest path between those two nodes; *(ii)* additional services chosen at random on all the services offered by the on-path nodes. The number of additional services is chosen uniformly at random between 1 and $N_{add} = 7$ which is a reasonable number for the sake of our simple evaluation. The units requested for each service $j$ $q^i_{S_j}$ are chosen uniformly at random between 1 and $Q_{max} = 30$, which is a number that can only be interpreted in comparison to the unit and maximal capacity of the services. The influence of the service capacity is shown later. The valuation of the bid $v_i$ (in $) is chosen uniformly at random between 1 and $\sum_{j\in[1..M]} q^i_{S_j}$, to account for the




(a) Operator revenue and total market valuation.




(b) % of accepted valuation.

Fig. 4: Effect of the number of bids on market properties with: $N_{VNF} = 5$, $C = 100$.

fact that, the more services are requested by a bid, the more his valuation has a chance to be higher. Since the system is incentive compatible, clients submit bid prices $b_i$ equal to their $v_i$.

### B. Heuristic vs. optimal allocation

Fig. 3 shows the difference between the total valuation computed by our heuristic and the optimal maximal sum of valuations found using CPLEX. The curves stop at 600 bids because the solver ran out of memory. The figure shows how the optimal and the approximation stay close together (less than 10% deviation) but seem to get farther as the number of bids grows. Note that we are assured that this approximate total valuation is in the worst case at least the square root of the optimal total valuation [19], but actually it is much closer to the optimal ($880 difference vs. a theoretical worst case of $9382 difference). We can conclude that the greedy heuristic performs well in our simulation environment, providing a good and fast approximation.

### C. Impact of the number of bids

We first show the importance of the number of bids on the operator revenue. Fig. 4a represents the market valuation and

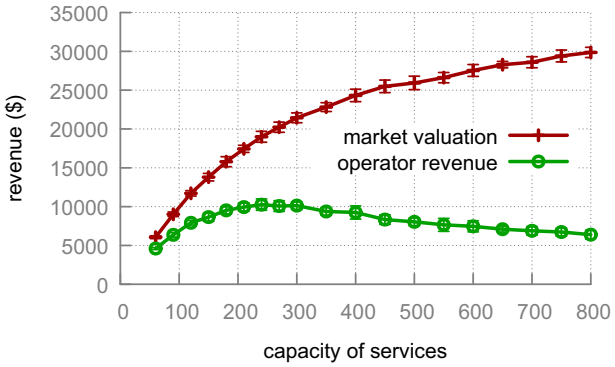
the operator revenue according to the number of bids. Since $b_i = v_i$, market valuation is defined as the sum of all accepted bid prices: $\sum_{i \in W} b_i$. A hundred bids are enough to use all available resources (about 20% of bids are rejected as seen in Fig. 4b), but the revenue and market valuation grows as we get more bids. This is because the more bids are submitted, the higher the probability of having high bids. We can also see that the revenue of the operator grows in accordance with the market valuation. When a high number of bids are submitted, a bigger part of market valuation is paid to the operator. Since the mechanism is incentive compatible, market valuation is equal to $\sum_{i \in W} v_i$. The operator revenue is the sum of all prices paid by the clients: $\sum_{i=1}^{N} p_i$. Thus, the difference between the two curves is actually the sum of the utilities of all participants $\sum_{i=1}^{N} u_i$. Operator revenue grows because the competition between the bidders grows. The price difference between a winning bid and the smallest possible winning bid price (the price actually paid by the client) gets smaller and smaller as the number of bids grows. Note that the right part of the curve represents a very competitive environment where only a small percentage of bids are accepted. Less than 10% of bids are accepted if there are more than 2000 bids, as shown in Fig. 4b.

### D. Impact of service capacity

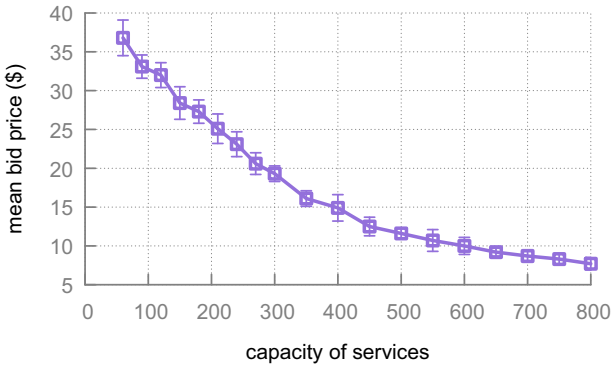
Fig. 5a shows what happens when the operator chooses to scale up the size of the offered services while demand does not change. The market value keeps growing because installing more capacity means that more bids can be accepted, even if the growth is not constant, since we start adding bids with a lower bid price. The revenue initially grows because there is a lot of competition and having more capacity means that more clients are accepted and pay the operator. However, adding capacity means that there is less and less competition for resources; hence the price paid by each winning bid gets smaller, as we can see in Fig. 5b. This is again because, with less competition, the minimal bid price for a winning bid $i$ to be accepted ($p_i$) gets smaller. The maximum revenue for the operator is in our case at a capacity of $C = 240$, with revenue of about $10,000. However, this does not consider the operator's cost of adding capacity, and depending on such cost the optimal might be anywhere between $C = 0$ and $C = 240$.

As we can see in Fig. 6 the optimal capacity is highly dependent on the demand level. During normal operations, leftover capacity will change from time slot to time slot on each PoP, so even assuming that demand is constant, revenue will change for the operator and it is not always possible to predict accurately the revenue that will be obtained on each time slot.

This simple model represents easily what might be the operator offers and clients bid, the general properties of the system are the same for a more complex model. The question of what is the optimal capacity to offer, and where should the capacity be placed in priority (as opposed to uniformly in our model) are important questions that the operator should answer when managing its NFV infrastructure and the auction.




(a) Operator revenue and total market valuation.




(b) Mean price paid by accepted bids.

Fig. 5: Effect of the capacity C of services offered by the operator on market properties with: $N = 1000$, $N_{VNF} = 5$.

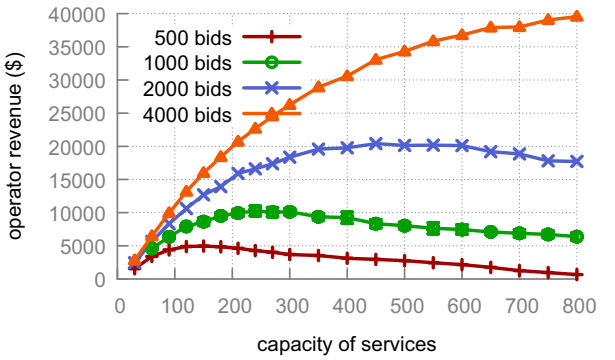



Fig. 6: Effect of capacity C on Operator revenue for 3 different number of bids with $N_{VNF} = 3$.

Note that here we only see what happens in a single time slot, whilst with dynamic demands the results will be different in each time slot.

## VII. Conclusions and Perspectives

In this paper we presented a novel game-theory approach for exploiting exceeding resources offering service function

chains in an NFV infrastructure that can be efficiently used to handle elastic network provisioning. The proposed approach offers a new business model and revenue opportunities to NFV operators. We proposed a Multi-Unit Combinatorial Auction system which accommodates clients by allowing them to directly express their service function chain demands as bids with a guaranty of obtaining the whole service function chain (and not isolated VNFs). The problem results being NP-Complete, but we proposed a heuristic that drastically simplifies the combinatorial nature of allocating bundles of resources, while guarantying that the approximation given is the best attainable in polynomial time. Moreover, our heuristic leads to an optimal economic efficiency and incentives clients to submit their true valuation. This is especially important in such a novel system since there is not yet good information on what exactly the utility of clients are, while operators cannot easily make assumptions on how they should price their services. Our evaluation results show that our heuristic gives a very good approximation of the economic efficiency (less than $10\%$ deviation from the optimum) and that the operator can keep a good share of the submitted bids even if the payment system is incentive compatible and lets clients pay a lower price than their submission. Our results also show the interesting trade off facing operators in this elastic VNF provisioning system, where the revenue generated is highly dependant on the quantity of available resources compared to the unpredictable demand.

Future work will include a dynamic study of the auctioning system to show how clients and operators can adapt to demands that span more than one time slot, with a more detailed model of VNF requirements and implementation considerations. We can then show how clients need to balance between a classical model of paying a higher price for long demands and the auction system for less critical NFV needs. We will also study the problem of revenue sharing for multiple NVF operators: In this paper, only a single operator offers VNFs but a client might need multiple operators to satisfy his needs. Independent auctions are not possible since a client might win one auction while losing in another operator domain. We then need coordination between operators and a fair way to allocate the price paid by the client for the whole network service chain to the multiple operators that are deploying the VNFs.

*Acknowledgments:* This work was partially supported by the French ANR Reflexion project (ANR-14-CE28-0019).

## REFERENCES

[1] GSNFV ETSI. Network functions virtualisation (NFV): Architectural framework. page V1.1.1, 2013.

[2] GSNFV ETSI. Network functions virtualisation (nfv); use cases. *V1.1.1*, Use case # 2:p15, 2013.

[3] Amazon ec2 spot instance, 2015. http://aws.amazon.com/ec2/purchasing-options/spot-instances/.

[4] Liang Zheng et al. How to bid the cloud. In *Proceedings of the 2015 ACM SIGCOMM*, pages 71–84.

[5] Anath Fischer, Juan Felipe Botero, Michael Till Beck, Hermann De Meer, and Xavier Hesselbach. Virtual network embedding: A survey. *Communications Surveys & Tutorials, IEEE*, 15(4):1888–1906, 2013.

[6] Peter Cramton, Yoav Shoham, and Richard Steinberg. Combinatorial auctions. 2006.

[7] Roger B Myerson. Incentive compatibility and the bargaining problem. *Econometrica: journal of the Econometric Society*, pages 61–73, 1979.

[8] Mathieu Bouet, Jérémie Leguay, Théo Combe, and Vania Conan. Cost-based placement of vdpi functions in nfv infrastructures. *International Journal of Network Management*, 25(6):490–506, 2015.

[9] Marcelo Caggiani Luizelli et al. Piecing together the nfv provisioning puzzle: Efficient placement and chaining of virtual network functions.

[10] Md Faizul Bari, Shihabur Rahman Chowdhury, Reaz Ahmed, and Raouf Boutaba. On orchestrating virtual network functions.

[11] Guiyi Wei, Athanasios V Vasilakos, Yao Zheng, and Naixue Xiong. A game-theoretic method of fair resource allocation for cloud computing services. *The journal of supercomputing*, 54(2):252–269, 2010.

[12] Ikki Fujiwara, Kento Aida, and Isao Ono. Applying double-sided combinational auctions to resource allocation in cloud computing. In *10th IEEE/IPSJSAINT 2010*, pages 7–14.

[13] P. Samimi and A. Patel. Review of pricing models for grid & cloud computing. In *ISCI, 2011 IEEE*, pages 634–639. IEEE, 2011.

[14] Rahul Jain and Jean Walrand. An efficient mechanism for network bandwidth auction. In *Network Operations and Management Symposium Workshops, 2008. NOMS Workshops 2008. IEEE*, pages 227–234. IEEE, 2008.

[15] Aurel A Lazar, Nemo Semret, et al. Design, analysis and simulation of the progressive second price auction for network bandwidth sharing. *Columbia University (April 1998)*, 1998.

[16] Sorabh Gandhi et al. A general framework for wireless spectrum auctions. In *2nd IEEE DySPAN 2007.*, pages 22–33.

[17] Stephen J Rassenti, Vernon L Smith, and Robert L Bulfin. A combinatorial auction mechanism for airport time slot allocation. *The Bell Journal of Economics*, pages 402–417, 1982.

[18] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V Vazirani. *Algorithmic game theory*, volume 1. Cambridge University Press Cambridge, 2007.

[19] R. Gonen and D. Lehmann. Optimal solutions for multi-unit combinatorial auctions: Branch and bound heuristics. In *EC'2,ACM*, pages 13–20, 2000.

[20] Kevin Leyton-Brown, Yoav Shoham, and Moshe Tennenholtz. An algorithm for multi-unit combinatorial auctions. In *AAAI/IAAI*, pages 56–61, 2000.

[21] Yair Bartal, Rica Gonen, and Noam Nisan. Incentive compatible multi unit combinatorial auctions. In *Proceedings of the 9th conference on TARK*, pages 72–87. ACM, 2003.

[22] Orna Agmon Ben-Yehuda, Muli Ben-Yehuda, Assaf Schuster, and Dan Tsafrir. Deconstructing amazon ec2 spot instance pricing. *ACM Transactions on Economics and Computation*, 1(3):16, 2013.

[23] Yuzo Fujishima, Kevin Leyton-Brown, and Yoav Shoham. Taming the computational complexity of combinatorial auctions: Optimal and approximate approaches. In *IJCAI*.

[24] Christos H Papadimitriou. *Computational complexity*. John Wiley and Sons Ltd., 2003.

[25] Uhlig et al. Providing public intradomain traffic matrices to the research community. *ACM SIGCOMM Computer Communication Review*, 36(1):83–86, 2006.