A Dynamic Pricing Algorithm for a Network of Virtual Resources.

Bram Naudts*, Mario Flores†, Rashid Mijumbi‡, Sofie Verbrugge*, Joan Serrat† and Didier Colle*
*Department of Information Technology
Ghent University, Ghent, Belgium
Email: {bram.naudts,sofie.verbrugge,didier.colle}@intec.ugent.be
†Network Engineering Department
Universitat Politècnica de Catalunya, Barcelona, Spain
Email: {mario.flores@entel,serrat@tsc}.upc.edu
‡Telecommunications Software and Systems Group
Waterford Institute of Technology, Waterford, Ireland
Email: rmijumbi@tssg.org

***Abstract*—A service chain is a combination of network services (e.g. network address translation (NAT), a firewall, etc.) that are interconnected to support an application (e.g. video-on-demand). Building a service chain requires a set of specialized hardware devices each of which need to be configured with their own command syntax. By moving management functions out of forwarding hardware into controller software, software-defined networking (SDN) simplifies provisioning and reconfiguration of service chains. By moving the network functions out of dedicated hardware devices into software running on standard x86 servers, network function virtualization (NFV) turns the deployment of a service chain into a more (cost)-efficient and flexible process. In an SDN/NFV-based architecture, those service chains are composed of virtual network functions (VNFs) that need to be mapped to physical network components. In literature, several algorithmic approaches exist to do so efficiently and cost-effectively. However, once mapped, a simple revenue model is used for pricing the requested substrate resources. This often leads to a loss of revenue for the infrastructure provider.**

In this paper, we propose a more advanced, dynamic pricing algorithm for pricing the requested substrate resources. The proposed algorithm increases the infrastructure provider's revenue based on historic data, current infrastructure utilization levels and the pricing of competitors. Our experimental evaluation shows that the proposed algorithm increases the revenue of the infrastructure provider significantly, independent of the average network utilization.

I. Introduction

Today, building a service chain to support a new application requires specialized hardware devices (middleboxes). Each middlebox needs to be configured individually via a vendor-specific syntax leading to complex configuration, high chance for error and high operational expenditures. As application loads vary over the day and often increase over time, building a service chain means overprovisioning of the devices to support the maximum level of demand leading to extra capital expenditures. As service chains are often built to support multiple applications, data sometimes passes through unnecessary network devices or servers.

By using SDN and NFV concepts, the effort and time needed to build a service chain to support a new application can be reduced. In a SDN/NFV-approach, a service chain refers to the abstraction used to define high-level services in a generic way. The service is described as a Service Graph (SG): a chain of high-level Network Functions (NFs) and pre-defined parameters.

These SGs need to be mapped to the physical infrastructure. As such, an embedding algorithm is needed to determine if the NFs and their connections in the SGs can be mapped to the physical infrastructure (virtual network embedding, VNE). In literature, several algorithmic approaches exist to solve the VNE problem. We refer the interested reader to [1] for a survey of VNE algorithms. Initial studies on placement of VNFs and VNF chains in both IP and optical networks are presented in [2], [3], [4], [5], [6], [7] and our own work [8].

There are however other, related challenges that receive fewer attention. For example, the authors of [9] declared that finding advanced economic models for VN pricing, instead of the simple revenue model used in the existing literature, is an important research topic that needs further attention. In that field, our work is related to the online pricing literature that deals with instantaneous demand dynamics and the adjustment of prices on the spot. Dynamic pricing has become an active field of the revenue management literature, with successful realworld applications in industries such as travel, fashion, etc. [10], [11], [12]. Closely related to our work, revenue management has also been applied to the field of cloud computing, [13], [14], [15], [16], [17]. For cloud providers, unlike other fields, revenue not only depends on the (unknown) number of customers, but also on the (unknown) duration of usage. As such, not only arrival rates but also service times are stochastic. In those works however, resources are considered as interchangeable. When embedding service graphs, the customer will however typically have a set of requirements (e.g. delay, location, etc.) for a network of resources. Resources are therefore hardly interchangeable without harming the expected quality-of-service. To our knowledge, work in this field is limited. The most related work to ours are [18] in which the negotiation process in a multi-domain environment

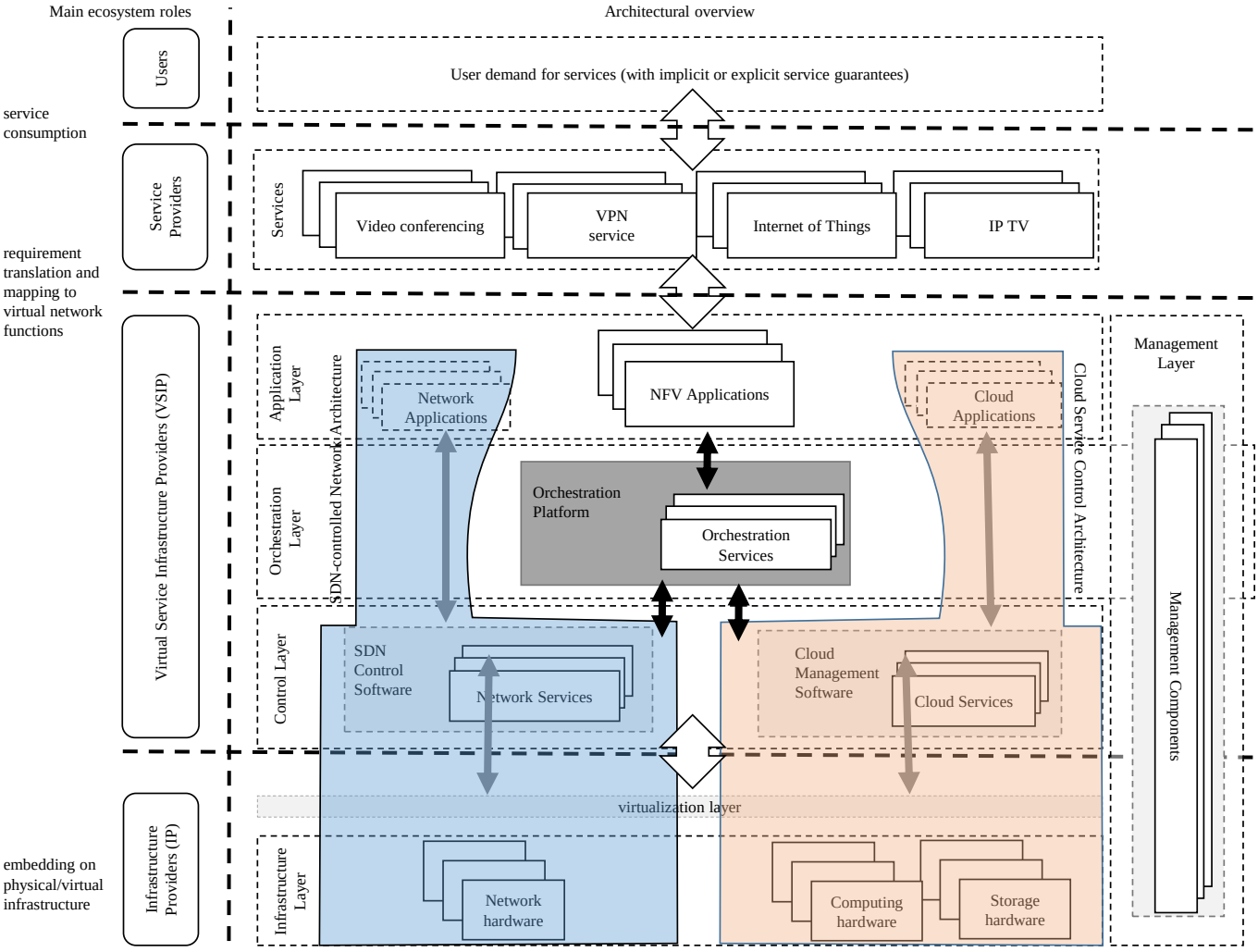

Fig. 1. Network and Cloud control architectures

is considered and [19] in which an auction based pricing strategies is used. These approaches are however dependent on a specific VNE algorithm, do not take into account the pricing strategy of competitors or they wait for a certain time to be able to batch a set of requests. To address this challenge, this paper proposes, a revenue management algorithm that deals with the problem of selling networks of virtualized, perishable resources to maximize the expected revenue from a population of price sensitive customers.

The remainder of this paper is organized as follows. Section II introduces the stakeholders and provides a detailed description of the problem. In section III, the proposed algorithm is introduced. The performance evaluation results are reported and discussed in section IV. Finally, section V concludes the paper.

II. Problem Description

A number of stakeholders are involved in the realization of an SDN/NFV-driven architecture for service chaining.

Ecosystem roles. On the left side of Figure 1, the most relevant ecosystem roles are represented. These roles are accomplished by the actors that actively participate in the exchange of value. Most actors will perform more than one role at the same time. For example, traditional ISPs fulfill the role of infrastructure provider, virtual service infrastructure provider and service provider.

Users. Users, i.e. end/enterprise users, retail or over-the-top providers, request and consume a diverse range of services. In general, users have no strong opinion about how the service is delivered as long as their quality of experience expectations are satisfied.

Service providers (SPs). SPs accommodate the service demand from users by offering one or multiple services including over-the-top services and X-play services (e.g. triple play). The service provider realizes the offered services on a (virtualized) infrastructure via the deployment of a SG composed of VNFs.

The decomposition of the SG into NFs is referred to as service decomposition. By decomposing a SG into elementary NFs, a number of benefits can be realized. First, re-usable

elementary blocks are developed. Second, new and more complex services can be realized from these elementary blocks and third, the detailed implementations of these NFs can be abstracted. Figure 2 depicts an example service decomposition. The service graph is decomposed into three NFs (NF1,NF2 and NF3), NF2 is decomposed to NF4 and NF5, etc.

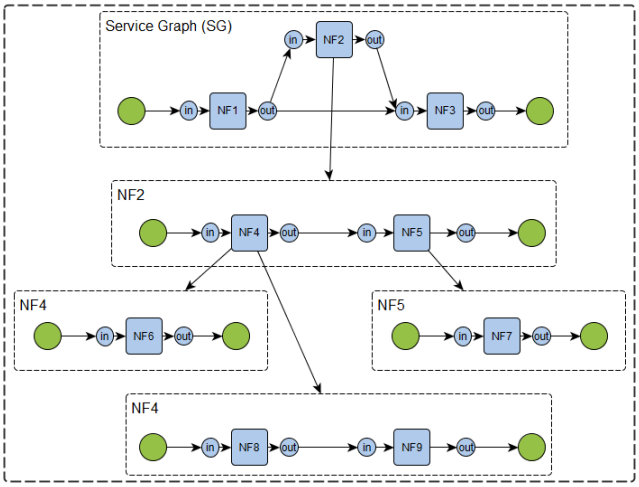

Fig. 2. Example of service decomposition process

These SGs are next handed to a virtual service infrastructure provider who will try to map the SG on the resources provided by the infrastructure provider.

Virtual service infrastructure providers (VSIPs). VSIPs deliver virtual service infrastructure to SPs, thereby meeting particular service level requirements by combining physical network and cloud resources owned by the IP into service infrastructure meeting particular SLA requirements [20].

A VNE algorithm should determine if the NFs and their connections in the SGs can be mapped to the infrastructure. To realize this, two control architectures are used to bring together the areas: (1) the software-driven[1] control of communication networks, and (2) the control of cloud (service) platforms. Both control architectures are depicted in the architectural overview of Figure 1 which is based on [21].

The first (in blue, left) is in charge of controlling the network of switching and routing equipment, the second (in orange, right) is in charge of creating and exposing cloud networks, i.e. a network of reusable computing and storage servers for the purpose of, e.g., building web services. The control architecture of both domains follows a roughly similar 3-layered approach.

At the lowest layer, infrastructure resources form the physical foundation on top of which services are provided. Communication networks rely on network hardware such as switches and routers, cloud infrastructures rely on (interconnected) computing and storage hardware (servers). These resources are owned by the infrastructure provider (see below).

A second layer, the control layer, interconnects the components of the infrastructure layer via their north-bound interface (e.g. OpenFlow for network control) in order to provide control-level services such as topology management or datastore services. Virtualization technology introduces a sublayer in between the infrastructure layer and the control layer, either at the device level, enabling one device to be segmented in multiple logical devices (e.g. in the case of server virtualization using Xen or Kernel-based Virtual Machine (KVM) virtualization technology), or at the level of multiple devices by horizontally segmenting or slicing an entire collection of devices (e.g. in the case of SDN-networks using FlowVisor).

At the highest layer, components of the application layer build further on control layer services to program client applications. A traffic engineering application might be defined on top of the SDN-control layer, while a Hadoop cluster might be an application on top of the cloud platform.

The orchestration platform has a complete view on available networking as well as on computing and storage resources and is used for services that require a combination of these resources. The orchestration components are able to make an informed decision on which infrastructure should be used. The provisioning process itself can then be further delegated to the already existing network and cloud control platforms.

Orthogonal to the horizontal layers, management functionality might be required to configure any of the components at the infrastructure, control or application layer for example to ensure policies or security-related options.

Infrastructure providers (IPs). IPs own and maintain the physical infrastructure and run the virtualization environments. By virtualizing the infrastructure, they open up their resources to remote parties for deploying VNFs. The reusable physical resources comprise all possible resource options (computing, storage and networking) and they span the entire service delivery chain from the end-user gateway and set-top-box over the access, aggregation and core network up to the cloud.

Negotiation process. The negotiation process considers the interaction between the SPs, VSIPs and the IPs. Their main objective is to maximize their own profit. Over time SPs send service request (Serv. Req.) to the VSIPs who try to embed the SG on the virtualized substrate resources of an IP.

Since provisioning of the SGs may be possible by multiple IPs, the VSIPs can use the competition between different IPs to negotiate the best price for a SG. VSIPs will as such use cost information (the prices charged by competing IPs) to cost-optimally solve the VNE problem. A service request (Serv. Req.) is sent by the VSIP to request for a mapping of a given service chain to the IPs. The request contains information about the amount of requested virtual resources and the duration. After receiving a SR, each IP attempts to perform a mapping. If the mapping is successful, the IP replies with a mapping proposal (MP) to the VSIP giving details of the mapping such as the price. If the mapping fails, the IP sends a mapping failed (MF) message. After receiving a MP,

[1]In the context of this article we focus on SDN-controlled networks, although traditional distributed routing protocols could also be considered as the control layer of communication networks.

the VSIP replies with an accept proposal (AP) to the IP with the best offer and with a reject proposal (RP) to all other IPs. The negotiation process is summarized in Figure 3.

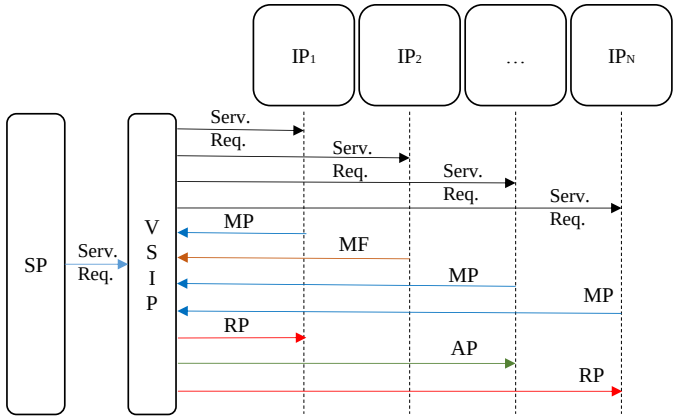

Fig. 3. Overview of the negotiation process

Objective. Our objective is to maximize the total revenue of the IP from a population of price sensitive customers (VSIPs). We therefore propose a revenue management mechanism based on a dynamic pricing algorithm. This dynamic pricing algorithm requires a set of inputs. Our assumption is that the IP records information about its own substrate resources (e.g. available and total resources, historic data about provided services per substrate resource) and about the pricing of its competitors (which can be obtained from public sources). We do not assume that the IP has information about the used VNE algorithm by the VSIP, nor about the about the state of the competitor's substrate resources.

III. Proposed Dynamic Pricing Algorithm

The goal of the infrastructure provider is to maximize its profit. One possible way to do so is to increase revenue. The infrastructure provider generates revenue by selling virtual resources (which are embedded on the physical infrastructure).

As discussed in the previous section, the customer (VSIP) favors the IP who is able to map the VNR at the lowest price. When we assume that n ($n \in [1, N]$) IPs provide an offer for virtual network request (VNR) v ($v \in [1, V]$) at a price $P_{n,v}$, the users willingness to pay for VNR v (WTP_v) is as such the minimum of the offers of the different IPs (Equation 1) with $P_{n,v}$ the price of IP n for virtual network request v.

$$WTP_v = min(P_{1,v}, P_{2,v}, P_{3,v}, \ldots, P_{N,v}) \tag{1}$$

As already mentioned in the previous section, we assume that the pricing of all other IPs is known for a VNR. The lowest price of all competitor is given by Equation 1.

Determining $P_{n,v}$. $P_{n,v}$ is typically composed of the units of substrate resource i ($i \in [1, I]$) demanded by VNR v ($R_{i,v}$), the price charged per unit of the substrate resources i (P_i) and the duration of the VNR v (D_v).

TABLE I
List of parameters and their symbols

parameter	symbol
infrastructure provider (IP)	n, $n \in [1, N]$
virtual network request (VNR)	v, $v \in [1, V]$
substrate resource (SR)	i, $i \in [1, I]$
constrained SR	c, $c \in [1, C]$
acceptance level of substrate resource i	a_i
price of IP n for VNR v	$P_{n,v}$
willingness to pay for VNR v	WTP_v
price of SR i, static, dynamic	$P_i, P_{i,s}, P_{i,d}$
price of VNR v, static,dynamic	P_v,$P_{v,s}$,$P_{v,d}$
units requested of substrate resource i by VNR v	$R_{i,v}$
duration of VNR v	D_v
expected revenue, at acceptance level a	$E(Rev)$,$E(Rev_a)$
the revenue per unit of SR i for VNR v	$RU_{i,v}$
the average revenue per unit of substrate resource i at the current level of acceptance	$\bar{RU}_{i,a}$
blocking probability of SR i at the current level of acceptance	$P_{b,i,a}$
the system ingress load	E
the mean arrival rate λ of SR i at acceptance level a	$\bar{\lambda_{i,a}}$
the mean service time of SR i	$\bar{s_i}$
the number of servers available for SR i at acceptance level a	$c_{i,a}$
overall acceptance level	A
discount rate	δ

$$P_{n,v} = \sum_{i=1}^{n} R_{i,v} \times P_i \times D_v \tag{2}$$

In static pricing, the parameter P_i of Equation 2 is a constant over time ($P_{i,s}$). In dynamic pricing, P_i evolves over time and as such the price per substrate resource i will be different per VNR ($P_{i,d}$). The goal of this research is to find an algorithm to determine $P_{i,d}$ in order to increase the total revenue of the infrastructure provider.

We summarize the used parameters in Table I and the proposed dynamic pricing approach in algorithm 1 and detail the algorithm below.

Determining $P_{i,d}$. The general idea of the algorithm is to vary the unit price of individual resources (e.g. substrate nodes or substrate links) based on the utilization level of the resource. If the utilization level is low (i.e. ample storage space, computing power or bandwidth available), there will be no constraint to embed several virtual network requests as they arrive one after the other. As such $P_{i,d}$ is set at a price below that of competitors to attract demand. However, when the utilization level of a resource is high, that resource may become constrained. As such, the IP will no longer be able to embed each VNR (e.g. only 9 out of every 10 VNRs). Therefore, when the utilization level of a substrate resource is high, that resource is protected and only those virtual network requests with a relatively high revenue per requested unit of the constrained substrate resource are priced attractively (in respect to the competitors prices) while those VNRs that have a relatively low revenue per requested unit of the constrained substrate resource are priced at a premium to compensate for the potential loss when a more valuable virtual

Algorithm 1 Dynamic pricing algorithm

while VNR arrive **do**
  map offer
  if offer can be mapped **then**
    gather historic data: WTP_v, D_v and $R_{i,v}$
    for all $i = 1$ to I **do**
      for all $v = 1$ to V **do**
        calculate $RU_{i,v}$ via Eq. 5
      end for
    end for
    order historic data in ascending order based on $RU_{i,v}$
    for all $i = 1$ to I **do**
      for all $a = \frac{v}{V}$ with $v \in [1, V]$ **do**
        calculate $\bar{RU}_{i,a}$ via Eq. 7
        calculate $c_{i,a}$ via Eq. 9
        calculate $P_{b,i,a}$ via Eq. 8
        calculate $E(Rev_a)$ via Eq. 6
      end for
    end for
    obtain WTP_v via Eq. 1
    for all $i = 1$ to I **do**
      select a corresponding with max $E(Rev_a)$
      calculate $P_{i,d}$ via Eq. 4
      calculate $P_{v,d}$ via Eq. 3
      if $P_{v,d} > WTP_v$ **then**
        add a_i to list of constrained resources
      end if
    end for
    if multiple constrained resources **then**
      apply algorithm 2
    end if
    for all $i = 1$ to I **do**
      calculate $P_{i,d}$ via Eq. 4
      calculate $P_{v,d}$ via Eq. 3
      save highest $P_{v,d}$ to P_v
    end for
    send offer to broker with price P_v
    if offer accepted **then**
      embed VNR
    end if
  else if VNR cannot be mapped **then**
    reject VNR
  end if
end while

network request would arrive which cannot be embedded due to resource shortage.

Instead of using Equation 2 for determining P_v, the price of the VNR is determined by multiplication of the dynamic price of the constrained substrate resource i ($P_{i,d}$) with the number of units requested of the constrained resource by VNR v ($R_{i,v}$) and the total duration of the VNR v (D_v). However, when P_v is lower than the P_v obtained by applying equation 1, the latter P_v is used (Equation 3).

$$P_v = max(P_{i,d} \times R_{i,v} \times D_v, Eq.1 \times (1-\delta)), \forall i \in \{1, I\} \quad (3)$$

In this pricing scheme, $P_{i,d}$ is dynamic and equal to $RU_{i,v=(1-a)\times V+1}$. $RU_{i,v=(1-a)\times V+1}$ is the revenue per unit of substrate resource i for VNR v for the acceptance level a with the maximum expected revenue ($E(rev)$). The level of acceptance refers to the number of VNRs that will be priced at an attractive price. For example, an acceptance level of 80% means that 8 out of every 10 VNRs will be priced attractively in respect to the competitors price while the other 2 will be priced at a premium (for which, the client will likely choose for a competitor unless there is no better option). Equation 4 is used to calculate $RU_{i,v=(1-a)\times V+1}$.

$$P_{i,d} = RU_{i,v=(1-a)\times V+1} = \frac{WTP_v}{\frac{D_v}{R_{i,v}}} \quad (4)$$

Determining a. To determine the acceptance level a with the maximum $E(rev)$, a database composed of historic data of all VNRs that have been mapped on substrate resource i is maintained. It contains three data fields: (1) the lowest price offered by all competitors for virtual network request v (WTP_v), (2) the duration of virtual network request v (D_v) and (3) the units requested of substrate resource i in virtual network request v ($R_{i,v}$). The ratio $RU_{i,v}$, the revenue per unit of substrate resource i for VNR v, is first calculated for each of the entries according to Equation 5.

$$RU_{i,v} = \frac{WTP_v}{\frac{D_v}{R_{i,v}}} \quad (5)$$

The database is next ordered in ascending order based on this ratio. Once ordered, the acceptance level a with the maximum $E(rev)$ can be found based on two factors: (1) the average revenue per unit of substrate resource i used at the current level of acceptance ($\bar{RU}_{i,a}$) and (2) the blocking probability of substrate resource i at the current level of acceptance ($P_{b,i,a}$).

$$E(Rev_a) = \bar{RU}_{i,a} \times (1 - P_{b,i,a}) \times a \quad (6)$$

Once $E(Rev_a)$ is determined for all a, the acceptance level a that corresponds with the the maximum $E(Rev)$ is chosen from the ordered list and used in Equation 4.

Determining $\bar{RU}_{i,a}$. The ordered database is used to calculate $\bar{RU}_{i,a}$. $\bar{RU}_i, a$ is then determined according to Equation 7.

$$\bar{RU}_{i,a} = \frac{\sum_{v=(1-a)\times V+1}^{V} \frac{\frac{WTP_v}{D_v}}{R_{i,v}}}{(V - v + 1)}, v \in [1, V], a = \frac{v}{V} \quad (7)$$

Determining $P_{b,i,a}$. To calculate $P_{b,i,a}$, the virtual network requests arriving at a substrate resource are modeled as a $M/M/c/K$ queuing system (Kendall's notation). This is a queuing system that needs to satisfy the conditions that arrivals form a single queue and arrive according to a Poisson process

(M, memoryless), that service times are exponentially distributed (second M), that there are c servers which serve from the front of the queue and a buffer capacity of K (including those in service). In an $M/M/c/K$ queue only K customers can queue at any one time (including those in service). Any further arrivals to the queue are considered lost for service. In this case, a substrate resource is either able to map a virtual network request or not. As such, the buffer size K is zero ($c = K$, $M/M/c/c$). The Erlang B formula (also known as the Erlang loss formula), can be used to calculate the blocking probability that describes the probability of losses for a group of identical parallel resources (Equation 8). In Equation 8, $c_{i,a}$ is the number of servers available for substrate resource i at acceptance level a and E is the systems ingress load in erlang which is calculated as the mean arrival rate λ of substrate resource i at acceptance level a ($\bar{\lambda_{i,a}}$) multiplied by the mean service time of substrate resource i ($\bar{s_i}$).

$$P_{b,i,a} = \frac{\frac{E^{c_{i,a}}}{c_{i,a}!}}{\sum_{j=0}^{c_{i,a}} \frac{E^j}{j!}} \tag{8}$$

$\bar{\lambda_{i,a}}$ is calculated by multiplying the historic arrival rate (the number of virtual network requests that arrive per time unit) with the acceptance level a. $\bar{s_i}$ is calculated as the mean service time of all virtual network requests that are mapped and use substrate resource i. To estimate $c_{i,a}$, the ordered database of historic data which is maintained per substrate resource is used in combination with the units available of the substrate resource i considered (A_i). $c_{i,a}$ can be calculated via Equation 9.

$$c_{i,a} = \lfloor \frac{\sum_{v=(1-a)\times V+1}^{V} \frac{A_i}{D_v}}{(V-v+1)} \rfloor, v \in [1,V], a = \frac{v}{V} \tag{9}$$

Once $c_{i,a}$ is known, $P_{b,i,a}$ can be determined via Equation 8. Once $R\bar{U}_{i,a}$ and $P_{b,i,a}$ are known, $E(Rev_a)$ can be calculated via Equation 6, etc.

Multiple constrained resources. When using the algorithm described above to price a VNR, multiple resources may be constrained instead of just one. When multiple resources are constrained, the overall acceptance level A will be lower than the acceptance level of each individual constrained substrate resource i (a_i). As a result, fewer VNRs will be attracted than initially hoped. To mitigate that risk, a_i is scaled by using Algorithm 2.

Algorithm 2 Scale availability

```
Ā ← (Σ_{i=1}^{C} a_i) / C
A* ← Π_{i=1}^{C} a_i
while Ā ≠ A* do
  S = ∛(Ā / A*)
  for all i such that 1 ≤ i ≤ C do
    a_i = min(1, S × a_i)
    if a_i > 1 then
      remove a_i from the list of constrained resources
    end if
  end for
  A* ← Π_{i=1}^{C} a_i
end while
```

IV. Performance evaluation

The focus of our evaluations is on quantifying the benefit of the revenue management algorithm in terms of total revenue. To ensure a fair comparison we model two identical IPs (i.e. same network topology and substrate capacity) who compete against each other. The first IP uses the dynamic pricing algorithm (Equation 3) while the second uses a static pricing algorithm (Equation 2).

A. Simulation setup.

We compare different simulation setups: (1) different inter-arrival rates ($\frac{1}{\lambda}$) and (2) different capacity requested per virtual node and link. The scenarios are summarized in Table II. Each setup is simulated for $20,000$ VNR arrivals. In the dynamic pricing algorithm, a discount δ of 5% is given (Equation 3).

TABLE II
Overview of the simulation input parameters

scenario	substrate node	virtual node	substrate link	virtual link	$\frac{1}{\lambda}$
1	100-200	10-20	200-400	16-40	1
2	100-200	10-20	200-400	16-40	2
3	100-200	10-20	200-400	16-40	3
4	100-200	10-20	200-400	16-40	5
5	100-200	25-50	200-400	40-100	1
6	100-200	25-50	200-400	40-100	2
7	100-200	25-50	200-400	40-100	3
8	100-200	25-50	200-400	40-100	5

Physical network. The physical infrastructure is modeled as an undirected graph. The infrastructure consists of nodes connected via links. Each node has certain capacity in terms of computation, memory and/or storage, each link has a certain capacity in terms of bandwidth and has a certain delay. The substrate network used in the simulations has 25 nodes and 75 links. The minimum and maximum capacity of each substrate node and link is given in Table II

Service request. Each service request is represented as a directed graph to support the dependency between elementary NFs. The NFs in are represented as nodes connected via directed links in the graph. Each NF has certain requirements in terms of computation, memory and/or storage and links connecting different NFs should meet certain requirement in terms of maximum allowed delay and bandwidth. The virtual networks used in the simulation have a maximum of 7 nodes and 12 links. The minimum and maximum capacity of each virtual node and link is given in Table II.

Virtual network embedding algorithm. The VNE algorithm is an implementation of a link-based multi-commodity flow formulation of the one-shot virtual network embedding [22] in CPLEX 12.6.

Negotiation process. The VNRs are awarded to the IP according to the negotiation process depicted in Figure 3 (we assume 2 IPs). The result of the negotiation process is classified in 5 categories: (1) F, the VNR cannot be mapped to either IP, (2) M1, only the first IP is able to map the request, (3) M2, only the second IP is able to map the request, (4) P1, both IPs are able to map and IP 1 has the best offer and (5) P2, both IPs are able to map and IP 2 has the best offer.

B. Results

We report the embedding results for each of the scenarios in Table III. As can be expected, when demand for substrate resources is high, e.g. due to a high interarrival time or/and VNRs that demand a large share of the substrate resources, the number of failed mappings is large and vice versa. Also, the number of VNRs that are won by the first provider by undercutting the competitor's price (P1) decrease when demand is high until both are more or less equal for very high levels of demand (e.g. simulation 5). This can be understood as (1) only very few requests can be mapped on the substrate network of both IPs and (2) the VNRs that receive a discount from IP 1 will be limited to those VNRs that have a high payoff per unit of the constrained resource.

TABLE III
EMBEDDING RESULTS FOR EACH SIMULATION SETUP.

simulation	M1	M2	P1	P2	F	total IP1	total IP2
1	15%	22%	13%	10%	40%	28%	32%
2	2%	26%	44%	18%	10%	46%	44%
3	0%	17%	68%	13%	2%	68%	30%
4	0%	7%	83%	9%	0%	83%	17%
5	9%	9%	1%	1%	80%	10%	10%
6	13%	17%	8%	4%	59%	21%	20%
7	15%	23%	13%	4%	44%	29%	27%
8	5%	28%	45%	13%	9%	50%	41%

When we focus on the total number of VNRs that each IP has obtained (its market share) it is clear that when demand is relatively slow (e.g. simulation 4), the first provider obtains the highest market share and also the highest total revenue, average node utilization and link utilization (Figure 4). This is reached by systematically undercutting the price of its competitors for those VNRs that are considered as valuable. It is however less obvious that IP1 is able to reach a higher total revenue than IP2 when its market share is lower (e.g. simulation 1). To clarify this we need to take into account the node and link utilization rates. These are higher even though the market share of the first IP is lower. The proposed revenue management model is able to obtain this result by pricing VNRs that have a high revenue per unit of the constrained substrate resources lower than its competitors while demanding a premium for those VNRs that have a low revenue per unit of the constrained substrate resources (Equation 3). The impact of this decision is further clarified in Table IV which presents the average revenue per VNR for each embedding result. By focusing on high value requests, the IP is able to increase its revenue per VNR. To do so, the IP needs to use its constrained resources optimally (certain substrate nodes and links) and at the same time reach a higher utilization rate for those resources that have a lower demand (e.g. substrate nodes with ample capacity).

TABLE IV
COMPARISON OF AVERAGE REVENUE PER VNR

simulation	average revenue P1	average revenue P2	average revenue M1	average revenue M2
1	93%	50%	100%	92%
2	98%	60%	100%	99%
3	100%	62%	96%	98%
4	100%	62%	98%	99%
5	100%	65%	99%	89%
6	92%	64%	100%	95%
7	97%	70%	100%	97%
8	100%	72%	-	72%

V. CONCLUSION

We have proposed an overarching SDN/NFV architecture which can be used to realize SGs of reusable network functions on a virtualized physical infrastructure owned by an infrastructure provider (IP). By mapping the most relevant ecosystem roles to the overarching SDN/NFV architecture, we illustrated the importance of revenue management models for the IP.

The revenue management model proposed in this paper has as goal to increase the total revenue of the IP. This goal is reached by a two-fold strategy. First, when demand is slow (low utilization), VNRs are attracted by underpricing competitors (via applying a discount). Second, when demand is strong, high value VNRs receive an offer that undercuts the price of competitors while low value VNRs are only embedded if a premium is paid. The value of a VNR is determined by those substrate resources that have a low amount of available capacity (constrained resources). The total value of the VNR divided by the units requested of the requested constrained resource determines the value of a VNR. By comparing that value against historic data, the VNR is classified as low or high value. The proposed algorithm carefully balances the extra revenue that can be gained by attracting high value VNRs versus the potential loss of a low value VNR to reach a higher total and average revenue for the IP. The algorithm has been validated via simulations and shows a clear improvement in terms of total revenue and average revenue per VNR.

In this work, the competitor's price is considered static over time. In future work, we will consider scenarios in which the competitor itself applies more advanced revenue management models.

ACKNOWLEDGMENT

This work is partly funded by the FP7 UNIFY (grant. no. 619609) and FLAMINGO, a Network of Excellence project (grant. no. 318488), supported by the European Commission under its Seventh Framework Programme.

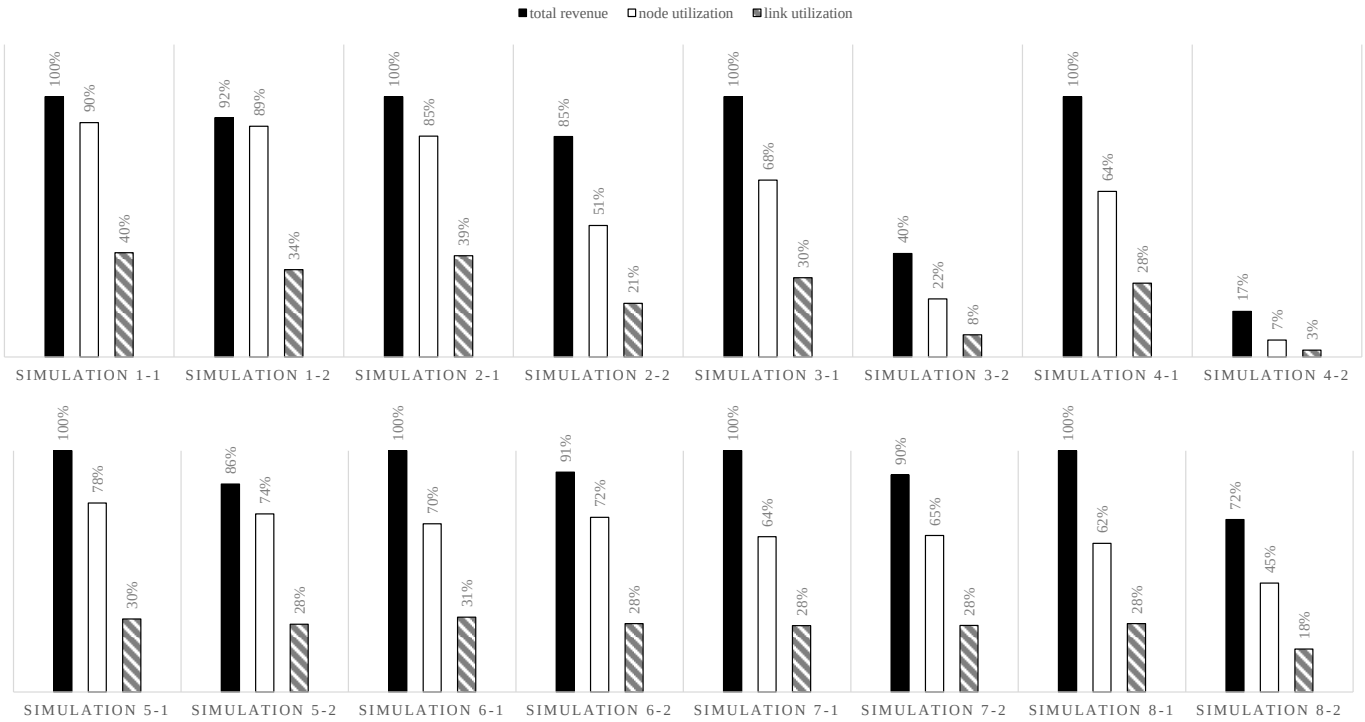

Fig. 4. Total revenue, average node and link utilization per simulation and per provider.

REFERENCES

[1] A. Fischer, J. F. Botero, M. Till Beck, H. De Meer, and X. Hesselbach, "Virtual network embedding: A survey," *Communications Surveys & Tutorials, IEEE*, vol. 15, no. 4, pp. 1888–1906, 2013.

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

[3] M. C. Luizelli, L. R. Bays, L. S. Buriol, M. P. Barcellos, and L. P. Gaspary, "Piecing together the nfv provisioning puzzle: Efficient placement and chaining of virtual network functions," *IFIP/IEEE IM*, 2015.

[4] H. Moens and F. De Turck, "Vnf-p: A model for efficient placement of virtualized network functions," in *Network and Service Management (CNSM), 2014 10th International Conference on*. IEEE, 2014, pp. 418–423.

[5] S. Sahhaf, W. Tavernier, M. Rost, S. Schmid, D. Colle, M. Pickavet, and P. Demeester, "Network service chaining with optimized network function embedding supporting service decompositions," *Computer Networks*, 2015.

[6] M. Xia, M. Shirazipour, Y. Zhang, H. Green, and A. Takacs, "Network function placement for nfv chaining in packet/optical datacenters," *Journal of Lightwave Technology*, vol. 33, no. 8, pp. 1565–1570, 2014.

[7] M. F. Bari, S. R. Chowdhury, R. Ahmed, and R. Boutaba, "On orchestrating virtual network functions," http://arxiv.org/abs/1503.06377, 2015, [Online; accessed 5-November-2015].

[8] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and S. Davy, "Design and evaluation of algorithms for mapping and scheduling of virtual network functions," in *Network Softwarization (NetSoft 2015), 2015 IEEE Conference on*. IEEE, 2015, pp. 1–9.

[9] N. Chowdhury, M. R. Rahman, and R. Boutaba, "Virtual network embedding with coordinated node and link mapping," in *INFOCOM 2009, IEEE*. IEEE, 2009, pp. 783–791.

[10] G. Bitran and R. Caldentey, "An overview of pricing models for revenue management," *Manufacturing & Service Operations Management*, vol. 5, no. 3, pp. 203–229, 2003.

[11] M. K. Geraghty and E. Johnson, "Revenue management saves national car rental," *Interfaces*, vol. 27, no. 1, pp. 107–127, 1997.

[12] B. C. Smith, J. F. Leimkuhler, and R. M. Darrow, "Yield management at american airlines," *interfaces*, vol. 22, no. 1, pp. 8–31, 1992.

[13] B. Javadi, R. K. Thulasiram, and R. Buyya, "Statistical modeling of spot instance prices in public cloud environments," in *Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on*. IEEE, 2011, pp. 219–228.

[14] H. Wang, Q. Jing, R. Chen, B. He, Z. Qian, and L. Zhou, "Distributed systems meet economics: pricing in the cloud," in *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*. USENIX Association, 2010, pp. 6–6.

[15] V. Kantere, D. Dash, G. Francois, S. Kyriakopoulou, and A. Ailamaki, "Optimal service pricing for a cloud cache," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 23, no. 9, pp. 1345–1358, 2011.

[16] H. Xu and B. Li, "A study of pricing for cloud resources," *ACM SIGMETRICS Performance Evaluation Review*, vol. 40, no. 4, pp. 3–12, 2013.

[17] ——, "Dynamic cloud pricing for revenue maximization," *Cloud Computing, IEEE Transactions on*, vol. 1, no. 2, pp. 158–171, 2013.

[18] R. Mijumbi, J.-L. Gorricho, J. Serrat, J. Rubio-Loyola, and R. Aguero, "Survivability-oriented negotiation algorithms for multi-domain virtual networks," in *Network and Service Management (CNSM), 2014 10th International Conference on*. IEEE, 2014, pp. 276–279.

[19] A. Jarray and A. Karmouch, "Vcg auction-based approach for efficient virtual network embedding," in *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on*. IEEE, 2013, pp. 609–615.

[20] S. Latre, J. Famaey, F. De Turck, and P. Demeester, "The fluid internet: service-centric management of a virtualized future internet," *Communications Magazine, IEEE*, vol. 52, no. 1, pp. 140–148, 2014.

[21] W. Tavernier, B. Naudts, D. Colle, M. Pickavet, and S. Verbrugge, "Can open-source projects (re-) shape the sdn/nfv-driven telecommunication market?" *it-Information Technology*, vol. 57, no. 5, pp. 267–276, 2015.

[22] R. Mijumbi, J. Serrat, J.-L. Gorricho, and R. Boutaba, "A path generation approach to embedding of virtual networks," *Network and Service Management, IEEE Transactions on*, vol. 12, no. 3, pp. 334–348, 2015.